

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-329256

(43) 公開日 平成8年(1996)12月13日

(51) Int.Cl.⁶

G 0 6 T 11/00

識別記号

庁内整理番号

9365-5H

F I

G 0 6 F 15/72

技術表示箇所

3 5 0

審査請求 未請求 請求項の数1 O L (全 49 頁)

(21) 出願番号 特願平8-124409

(22) 出願日 平成8年(1996)5月20日

(31) 優先権主張番号 4 6 9, 9 7 5

(32) 優先日 1995年6月6日

(33) 優先権主張国 米国 (U S)

(71) 出願人 590000400

ヒューレット・パッカード・カンパニー
アメリカ合衆国カリフォルニア州パロアル
ト ハノーバー・ストリート 3000

(72) 発明者 パイロン・エイ・アルコーン

アメリカ合衆国80526コロラド州フォー
ト・コリンズ、ベンサベン・ストリート
3931

(72) 発明者 ダレル・エヌ・エモット

アメリカ合衆国80526コロラド州フォー
ト・コリンズ、モス・クリーク・ドライブ
3931

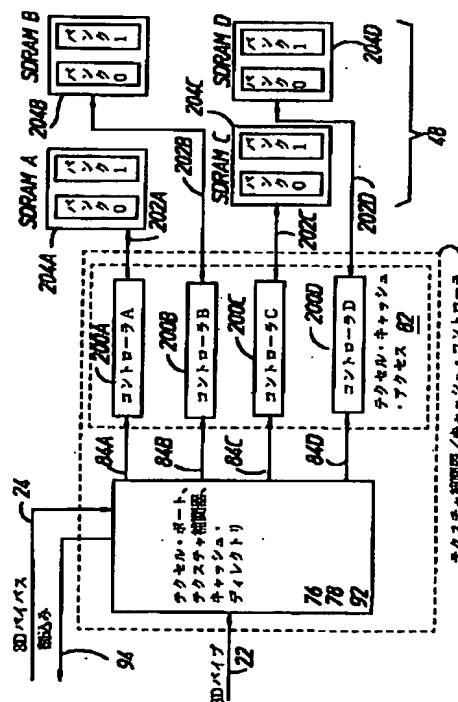
(74) 代理人 弁理士 岡田 次生

(54) 【発明の名称】 テクスチャ・データ割り当て方法

(57) 【要約】

【課題】 テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、テクスチャ・データを割り当て記憶する方法および装置を提供する。

【解決手段】 テクスチャ・データは、少なくとも一連のテクスチャMIPマップを含む。各MIPマップを少なくとも2つの部分に分割し、それらマップ部分を等サイズの複数のデータ・ブロックに割り当てる。マップ部分のサイズは1つのデータ・ブロックより小さい。上記ブロックはシステムの主メモリに記憶され、必要に応じて、システムのローカル・メモリへ1回に少なくとも1ブロックずつダウンロードされる。ダウンロードの際、各ブロックは、ローカル・メモリを構成する少なくとも1つのSDRAMの第1または第2のバンクの1つに記憶される。



【特許請求の範囲】

【請求項 1】 テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、個別にアクセス可能な少なくとも第 1 および第 2 のメモリ領域にテクスチャ・データを割り当てる方法であって、

少なくとも一連のテクスチャ MIP マップを、等しいサイズの複数のデータ・ブロックに分割するステップと、上記少なくとも一連のテクスチャ MIP マップの中の 1 つおきの MIP マップに含まれる共通のテクスチャ・データを含む第 1 のブロックを上記第 1 のメモリ領域に記憶するステップと、

上記第 1 のブロックに含まれるテクスチャ・データおよび上記 1 つおきの MIP マップに隣接する MIP マップに共通に含まれる共通のテクスチャ・データを含む第 2 のブロックを上記第 2 のメモリ領域に記憶するステップと、を含む方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、一般的に、テクスチャ・マッピングを行うコンピュータ・グラフィックス・システムに関するもので、特にテクスチャ・マッピング・データを記憶するキャッシュ・メモリ・システムに関するものである。

【0002】

【従来の技術】 コンピュータ・グラフィックス・システムは、2次元表示画面上でオブジェクトのグラフィック表現を表示するために一般に使用される。現在のコンピュータ・グラフィックス・システムは、高度に細密な表現を提供することができ、種々のアプリケーションにおいて使用されている。

【0003】 典型的なコンピュータ・グラフィックス・システムにおいては、表示画面に表現されるべきオブジェクトは、複数のグラフィックス・プリミティブに分解される。プリミティブは、グラフィックス・ピクチャの基本コンポーネントであって、点、線、ベクトルおよび三角形のような多角形を含む場合がある。典型的ハードウェア/ソフトウェア方式は、画面上に表現される 1 つまたは複数のオブジェクトの画像を表現するグラフィックス・プリミティブを、2次元表示画面上にレンダリング(rendering)または描画するように実施される。

【0004】 レンダリングされるべき 3次元オブジェクトを定義するプリミティブは、典型的には、ホスト・コンピュータによってプリミティブ・データとして定義され提供される。例えば、あるプリミティブが三角形であるとすれば、ホスト・コンピュータは、三角形の各頂点の x, y, z 座標および各頂点の R, G, B カラー値として、そのプリミティブを定義する。レンダリング・ハードウェアは、プリミティブ・データを補間して、各ピクセルを表現するため画面上オンにする表示画面ピクセル

ルおよび各ピクセルに関する R, G, B 値を計算する。

【0005】 初期のグラフィックス・システムは、複雑な 3次元のオブジェクトを表現またはモデル化する場合十分に現実的な形態で画像を表示することができなかった。そのようなシステムによって表示される画像は、極端になめらかな表面をしていて、モデル化されたオブジェクトに存在するテクスチャ、凹凸、スクラッチ、陰影およびその他の表面細部に欠けていた。

【0006】 このため、表面細部が改善された画像を表示するいくつかの方法が開発された。テクスチャ・マッピング手法はそのような方法の 1 つであって、これは、先ず、テクスチャ(textureすなわち表面模様)と呼ばれるソース画像を 3次元オブジェクトの面にマッピングし、次に、その 3次元オブジェクトを 2次元グラフィックス表示画面にマップして、その結果生成される画像を表示する。一般にマップされたテクスチャの細部の面特性は、カラー、鏡面反射、ベクトル摂動、反射、透明性、陰影、表面不規則性および階調度を含む。

【0007】 テクスチャ・マッピングは、テクスチャの 1 つまたは複数のテクセル(texelすなわち点要素)を、テクスチャがマップされる先のオブジェクトの表示部分の点要素(すなわちピクセルまたは画素)に対応させることを意味する。テクスチャ・マッピング・ハードウェアは、テクスチャ・マップにおけるテクセルがオブジェクトを表現する表示画面のピクセルに対応する形態を標示する情報を伝統的に備えている。テクスチャ・マップにおける各テクセルは、2次元テクスチャ・マップにおけるその位置を識別する S および T 座標によって定義される。各ピクセル毎に、該ピクセルに対応する 1 つまたは複数のテクセルが、テクスチャ・マップから取り出され、表示画面上でテクスチャ化されたオブジェクトを表現するため該ピクセルに対して生成される最終的 R, G, B 値と統合される。

【0008】 オブジェクト・プリミティブのピクセルの各々は、オブジェクトのあらゆる表現についてテクスチャ・マップにおける単一のテクセルと 1 対 1 の対応関係でマップすることはできない点理解されるべきである。例えば、オブジェクトが表示画面上で表示される場合表示位置に接近すればする程、オブジェクトは一層大きく表現される。オブジェクトが表示画面上大きく表現される程、テクスチャの表現はより詳細になる。従って、オブジェクトが表示画面の大部分を使う時、オブジェクトを表示画面で表現するため多数のピクセルが使用され、オブジェクトを表現する各ピクセルは、テクスチャ・マップにおける 1 つのテクセルと 1 対 1 の対応関係をもってマップされるか、あるいは、単 1 のテクセルが複数のピクセルに対応することもある。しかし、逆にオブジェクトが表示画面の相対的に小さい部分を占める時、オブジェクトを表示画面で表現するため相対的に少ないピクセルが使用され、テクスチャの表現は粗くなり、従っ

て、各ピクセルは複数のテクセルに対応することになる。テクスチャがオブジェクトの小さい部分にマップされる場合、各ピクセルは、また、複数のテクセルにマップされる可能性がある。この結果、典型的には、ピクセルの各々毎に、複数のテクセルに対応し、かつ、そのピクセルへ対応するテクセルの平均を表すテクセル・データが計算される。

【0009】典型的テクスチャ・マッピング・ハードウェア・システムは、レンダリングされるオブジェクトに関連するテクスチャを表現するデータを記憶するローカル・メモリを含む。上述のように、1つのピクセルが複数のテクセルに対応する場合がある。平均値を生成するためテクスチャ・マッピング・ハードウェアがローカル・メモリから1つのピクセルに対応する多数のテクセルを読み取らねばならないとすれば、多数のメモリ読み出し動作と多数のテクセル値の平均算出演算が必要となり、これは時間浪費的であってシステム処理能力を低下させる原因となるであろう。

【0010】この問題を解決するため、各テクスチャ毎に一連のMIPマップを作成し、レンダリングされるテクスチャのMIPマップをテクスチャ・マッピング・ハードウェアのローカル・メモリに記憶する方式が開発された。あるテクスチャに関するMIPマップは、テクスチャ・マップに直接対応する基本マップならびにそれ以外の一連のフィルタされたマップを含み、この場合、連続するマップは前後で、2の自乗のサイズで減少する。図1は1組のMIPマップの例を示す。(注：MIPは、multum in parvoの頭文字をとったもので、「小さい場所の多数の事柄」を意味する)。図1のMIPマップは、サイズの8×8テクセルである基本マップ100の他、それぞれサイズ4×4、2×2ならびに1×1テクセルである一連のマップ102、104および108を含む。

【0011】4×4マップ102は、基本マップ100をフィルタ(すなわち比例減少)することによって生成されるが、具体的には、マップ102のテクセルの各々は、基本マップ100の4個のテクセルの平均値に対応する。例えば、マップ102のテクセル110は、マップ100のテクセル112ないし115の平均に等しく、マップ102のテクセル118および120は、マップ100のテクセル121ないし124の平均およびテクセル125ないし128の平均にそれぞれ等しい。2×2マップ104は、例えばマップ104のテクセル130がマップ102のテクセル110、118、119および120の平均に等しくなるように、マップ102をフィルタすることによって同様に生成される。マップ108の単一(1×1)テクセルは、マップ104の4つのテクセルを平均することによって生成される。

【0012】従来技術のグラフィックス・システムは、一般的に、ホスト・コンピュータの主メモリからテクス

チャ・マッピング・ハードウェアのローカル・メモリへ、表示画面にレンダリングされるプリミティブに関して使用されるべきテクスチャに対する完全な一連のMIPマップをダウンロードする。従って、テクスチャ・マッピング・ハードウェアは、一連のMIPマップのうちのいずれのマップからもテクスチャ・データをアクセスすることができる。特定のピクセルについてテクセル・データを提供するためどのマップにアクセスすべきかは、当該ピクセルがマップするテクセルの数に基づいて決定される。例えば、ピクセルがテクスチャ・マップの単一のテクセルと1対1の対応関係でマップする場合は、基本マップ100がアクセスされる。しかし、ピクセルが、4、16または64のテクセルへマップするとすれば、マップ102、104および108が、それぞれテクスチャ・マップの4、16、64個のテクセルの平均を表すテクセル・データを保持しているので、それらのマップがそれぞれアクセスされる。

【0013】1つのピクセルが選択されたマップのどの1つのテクセルにも直接対応せず、複数のテクセルの間に落ちることがある。このような場合、グラフィックス・システムによっては、テクセル・データを正確に生成するため双線形補間(bilinear interpolation)を使う。1つのピクセルが、1つのMIPマップの複数のテクセル・エントリに対応する場合、使用されるテクセル・データは、最も近いテクセル・エントリの加重平均である。このように、ピクセルに対応するテクセル・データは、単一のマップにおける4つのテクセル・エントリの加重平均とすることができる。例えば、ピクセルがマップ102の132で示される位置に対応する場合、そのピクセルに対応するテクセル・データは、テクセル110、118、199および120の加重平均である。

【0014】また、ピクセルが一連のMIPマップのどのマップにも直接対応せず、2つのマップの間に落ちることもある。例えば、あるピクセルが、テクスチャ・マップの1より大で4未満の数のテクセルに対応することもある。このような場合、所望のテクセル・データを作成するため、グラフィックス・システムによっては、2つの近接するMIPマップの間で補間を行う。例えば、上述のように、1つのピクセルがテクスチャ・マップの1より大で4未満の数のテクセルに対応するような場合、マップ100および102によって提供されるテクセル・データが補間され、該ピクセルに関するテクセル・データが作成される。単一のマップにおける複数のテクセル・エントリの上述の補間と組み合わせられる時、この方式は、3線形補間(trilinear interpolation)として知られるもので、2つの近接するマップのそれぞれにおける4個の近接するテクセル、すなわち8個のテクセルの加重平均として特定のピクセルに対する所望のテクセル・データを生成することができる。

【0015】

【発明が解決しようとする課題】 上述のように、従来技術のテクスチャ・マッピング・システムは、システムによってレンダリングされるべきプリミティブに関連するいかなるテクスチャについても、一連のMIPマップを（たとえ一部がアクセスされないとしても）すべてダウンロードする。アクセスされないMIPマップのダウンロードは、アクセスされるマップのうちの使用されない部分のダウンロードと共に、システム資源の浪費であり、その帯域幅（すなわちデータ伝送率）を減少させる。

【0016】 更に、種々の動作が異なるオブジェクト・プリミティブに関して同時に実行されるように、パイプライン形態を実施するテクスチャ・マッピング・システムもある。しかし、そのようなシステムでは、1つのテクスチャに関する一連のMIPマップが非常に大きくなる可能性がある。大部分のシステムが使用するローカル・メモリは、そのような大規模のMIPマップを1度に1つしか記憶することができない。従って、プリミティブをレンダリングする際テクスチャの切り替えが行われる場合、システムは新たな一連のMIPマップをダウンロードしなければならない。新しいテクスチャ・データをテクスチャ・マッピング・ハードウェアのローカル・メモリにロードするために使用されるデータ経路は、典型的には、システムのプリミティブ・レンダリング・パイプラインを通過する。従って、新しいテクスチャをマップする必要がある場合、新たな一連のMIPマップがダウンロードされる前にプリミティブ・レンダリング・パイプラインをフラッシュ(flush)することができなければならない。一連のMIPマップがダウンロードされたなら、パイプラインに再びデータを送らなければならない。このように、新しいテクスチャが必要とされる度毎にプリミティブ・レンダリング・パイプラインをフラッシュしなければならないので、システムの帯域幅が低下する。

【0017】

【課題を解決するための手段】 本発明の1つの実施形態に従って、テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、個別にアクセス可能な少なくとも第1および第2のメモリ領域にテクスチャ・データを割り当てる方法が提供される。該方法は、少なくとも一連のテクスチャMIPマップを、等しいサイズの複数のデータ・ブロックに分割するステップ、上記少なくとも一連のテクスチャMIPマップの中の1つおきのMIPマップに含まれる共通のテクスチャ・データを含む少なくとも第1のブロックを上記第1のメモリ領域に記憶するステップ、および上記第1のブロックに含まれるテクスチャ・データおよび上記1つおきのMIPマップに隣接するMIPマップに共通に含まれる共通テクスチャ・データを含む第2のブロックを上記第2のメモリ領域に記憶するステップを含む。上記分割するステッ

プは、少なくとも一連のテクスチャ・マップの各マップを少なくとも2つの部分に分割するステップを含み、更に、そのマップ部分を複数の等サイズのブロックに割り当てるステップを含む。該方法は、更に、ブロックに記憶された特定のマップ、ブロックに記憶されたマップ部分、およびブロックに記憶されたマップのテクスチャをそれぞれ識別するため、各ブロックにブロック・タグを割り当てるステップを含む。

【0018】 本発明の1つの実施形態では、上記分割するステップが、更に、各ブロックを複数の等サイズのサブテクスチャに小分割するステップを含む。この実施形態において、上記方法は、更に、対応するブロック内のサブテクスチャの位置を識別するサブテクスチャ識別子を各サブテクスチャに割り当てるステップを含む。

【0019】 本発明の別の1つの実施形態では、テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、少なくとも一連のテクスチャMIPマップを含むテクスチャ・データを割り当て、記憶する方法が提供される。該方法は、上記少なくとも一連のテクスチャ・マップの各マップを少なくとも2つのマップ部分に分割するステップ、1つのブロック内に割り当てられるマップの部分が1つのブロックのサイズより小さくなるように各マップの部分を複数の等サイズのデータ・ブロックに割り当てるステップ、上記複数のデータ・ブロックをシステムの主メモリ内に記憶するステップ、および1回に少なくとも1ブロックずつ主メモリからシステムのローカル・メモリへ上記ブロックをダウンロードするステップを含む。1つの実施形態においては、上記ダウンロードするステップが、各ブロックを、少なくとも1つのSDRAMの第1または第2のバンクの1つにダウンロードするステップを含む。

【0020】 本発明の別の1つの実施形態では、テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、少なくとも一連のMIPマップを含むテクスチャ・データのブロックを記憶するためのメモリが提供される。上記メモリの第1の領域が、上記少なくとも一連のテクスチャMIPマップの中の1つおきのMIPマップに含まれる共通のテクスチャ・データを含む少なくとも第1のブロックを記憶する。上記メモリの第2の領域が、上記第1の領域とは別個にアクセスすることが可能であって、上記第1のブロックに含まれるテクスチャ・データおよび上記1つおきのMIPマップに隣接するMIPマップに共通に含まれる共通のテクスチャ・データを含む少なくとも第2のブロックを記憶する。1つの実施形態において、上記メモリは少なくとも1つのSDRAMを含み、上記第1および第2の領域は少なくとも1つのSDRAMの第1および第2のバンクをそれぞれ含む。本発明の1つの実施形態において、上記メモリは、各々が少なくとも1つのSDRAMを含む個別にアクセス可能な複数のインターリーブを更に含む。

【0021】

【発明の実施の形態】

1. システム概要

図2は、テクスチャ・データをローカルに記憶するためのキャッシュ・メモリを備えるテクスチャ・マッピング・ハードウェアを含む本発明のグラフィックス・システムの1つの実施形態のブロック図である。図示されている実施例が、基板ならびにチップの数、細分化の形態、バス幅およびデータ転送速度に関して単なる典型例にすぎないことは理解されるべきである。図示される以外のその他多数の形態を実施することは可能である。図2に示されているように、本システムは、フロントエンド基板10、テクスチャ・マッピング基板12およびフレーム・バッファ基板14を含む。フロントエンド基板は、52ビット幅バス16を経由してホスト・コンピュータ15と通信する。フロントエンド基板は、レンダリング（描画）すべきプリミティブを、バス16経由でホスト・コンピュータから受け取る。プリミティブは、x, y, zベクトル座標データ、R, G, Bカラー・データおよびテクスチャS, T座標によって指定される。これらデータは、すべて、例えばプリミティブが三角形であれば頂点のようなプリミティブの部分に関するデータである。次に、プリミティブを3次元で表すデータが、フロントエンド基板10によって、テクスチャ・マッピング基板12およびフレーム・バッファ基板14に対して85ビット幅バス18を経由して与えられる。テクスチャ・マッピング基板は、プリミティブを表現する画面表示ピクセルを計算するため受け取ったプリミティブ・データを補間し、該プリミティブのピクセル毎に対応するテクスチャ・データを決定する。計算結果のテクスチャ・データは、図2では単純化のため1本線として示されている5本の55ビット幅バス28を経由してフレーム・バッファ基板に送られる。

【0022】フレーム・バッファ基板14もまた、フロントエンド基板10から受け取ったプリミティブ・データを補間し、各プリミティブを表現する表示画面上のピクセルを計算し、各ピクセル毎にオブジェクトのカラーを決定する。次に、フレーム・バッファ基板は、上記オブジェクト・カラー値をテクスチャ・マッピング基板から送られたテクスチャ・データとピクセル毎に結合して、各ピクセル毎に画像R, G, Bを生成する。表示画面（図示されてない）のピクセルを制御するため各ピクセルに関するR, G, Bカラー制御信号が、R, G, Bライン29経由でそれぞれ与えられ、テクスチャ・マップ・プリミティブを表現する画像が表示画面上に表示される。

【0023】フロントエンド基板10、テクスチャ・マッピング基板12およびフレーム・バッファ基板14の各々はパイプライン化され、複数のプリミティブに対して同時に動作する。テクスチャ・マッピングおよびフレ

ーム・バッファ基板が、フロントエンド基板によって前に提供されたプリミティブに対して動作を行う際、フロントエンド基板は、基板12および14のパイプラインがいっぱいにならない限り、新しいプリミティブに対して動作し基板12および14へ提供し続ける。フロントエンド基板10は、分配器チップ30、3次元（3D）加速器チップ32A、32Bならびに32C、2次元（2D）加速器チップ34および集線器チップ36を含む。分配器チップ30は、X, Y, Z座標およびカラー・プリミティブ・データをバス16経由でホスト・コンピュータから受け取り、3次元プリミティブ・データを3次元加速器チップ32A、32Bおよび32Cに均等に分配する。このような形態で、3つのグループのプリミティブが同時に処理されることによって、システムの帯域幅が増加される。データは、40ビット幅バス38Aを経由して3次元加速器チップ32Aおよび32Bに送られ、40ビット幅バス38Bを経由してチップ32Cに送られる。バス38Aおよび38Bは、60MHzの伝送率でデータを伝送し、2つの3次元加速器チップをサポートするために十分な帯域幅を提供する。2Dプリミティブ・データは、44ビット幅バス40を経由して40MHzの伝送率で2D加速器チップ34へ送られる。

【0024】3次元加速器チップの各々は、受け取ったプリミティブを定義するx, y, z座標を、対応する画面空間座標に変換し、画面空間座標に対するオブジェクトR, G, B値およびテクスチャS, T値を決定し、プリミティブの四辺形を三角形へ分解し、各三角形を定義するため三角形平面方程式を計算する。各3次元加速器チップは、また、複数ウィンドウが表示される時、あるいは、プリミティブの一部が表示画面上に表される視野を越えて広がる時、画像の正確な画面表示を確実にするため視野クリッピング動作を実行する。3次元加速器32Aおよび32Bからの出力データは44ビット幅バス42Aを経由して、また3次元加速器32Cからの出力データは44ビット幅バス42Bを経由して、それぞれ、集線器チップ36へ60MHzの伝送率で送られる。2次元加速器34からの出力データは46ビット幅バス44を経由して集線器チップ36へ40MHzの伝送率で送られる。集線器チップ36は、3次元加速器チップ32A-32Cから受け取った3次元プリミティブ出力データを結合し、分配器チップ30による分配の前の元の順序にプリミティブを配列し直し、結合したプリミティブ出力データをバス18を経由してテクスチャ・マッピング基板およびフレーム・バッファ基板に送る。

【0025】テクスチャ・マッピング基板12は、テクスチャ・マッピング・チップ46、および、好ましくはキャッシュ・メモリとして構成されるローカル・メモリ48を備える。本発明の1つの好ましい実施形態において、ローカル・メモリは、後述の理由から、複数のSD

RAMチップ（すなわち同期ダイナミックRAM）から形成される。詳細は後述するが、キャッシュ・メモリ48は、フレーム・バッファ基板においてレンダリングされるプリミティブに関連するテクスチャMIPマップ・データを記憶する。テクスチャMIPマップ・データは、ホスト・コンピュータ15の主メモリ17から、バス40を経由して、2D加速器チップ34を通過し、24ビット幅バス24を経由して、キャッシュ・メモリ48にダウンロードされる。

【0026】テクスチャ・マッピング・チップ46は、表示画面上で描画（レンダリング）されるべきプリミティブを表すプリミティブ・データをバス18経由で連続的に受け取る。上述のように、3次元加速器チップ32A-32Cから送られるプリミティブは、点、線分および三角形を含む。テクスチャ・マッピング基板は、点または線分に関してはテクスチャ・マッピングを実行せず、三角形プリミティブについてのみ実行する。三角形プリミティブを表現するデータは、少なくとも1つの頂点に関するx, y, zオブジェクト・ピクセル座標、少なくとも1つの頂点のオブジェクト・カラーR, G, B値、少なくとも1つの頂点に対応するテクスチャ・マップ部分のS, T座標、および三角形の平面方程式を含む。テクスチャ・マッピング・チップ46は、オブジェクト・ピクセルz座標およびオブジェクト・カラーR, G, B値を無視する。チップ46は、x, yピクセル座標を補間し、プリミティブを表現する各x, y画面表示ピクセルに対応するSおよびT座標を補間する。各ピクセル毎に、テクスチャ・マッピング・チップは、ピクセルに対応するテクスチャMIPマップ部分をキャッシュ・メモリから取り出し、複数のテクセルの加重平均を含むテクスチャ・データを該ピクセルについて計算する。

【0027】1つの典型的実施形態において、キャッシュは1ブロックが256×256テクセルからなる64ブロックのテクセルを記憶する。従来技術のシステムのテクスチャ・マッピング・ハードウェアで使われるローカル・メモリと異なって、本発明のキャッシュ・メモリは、レンダリングされるプリミティブに対応する（大規模な）テクスチャの一連のMIPマップ全体を記憶しなくてもよい。むしろ、本発明のキャッシュ・メモリは、ある1時点をとると、その時点でプリミティブをレンダリングする場合一連のMIPマップの実際に使用される特定部分のみを記憶する。従って、ほとんどのアプリケーションの場合、ある1時点で、全テクスチャ・データのうちレンダリングされる画像に関する部分だけがキャッシュ・メモリに記憶される。

【0028】各テクスチャに関する完全な一連のMIPマップは、ホスト・コンピュータ15の主メモリ17に記憶される。レンダリングされるプリミティブの各ピクセルについて、テクスチャ・マッピング・チップ46は、キャッシュ・メモリ48のディレクトリにアクセス

して、テクスチャMIPマップの対応する1つまたは複数のテクセルが現在キャッシュに存在するかどうかを判断する。対応するテクセルがアクセス時点でキャッシュ・メモリに存在する場合、キャッシュ・ヒット(cache hit)が発生し、テクセルがキャッシュ・メモリから読み取られ、テクスチャ・マッピング・チップ46によってフレーム・バッファ基板に渡されるテクスチャ・データが計算される。

【0029】しかし、プリミティブ・ピクセルについて対応するテクセルがアクセス時点で存在しない場合、キャッシュ・ミス(cache miss)が発生する。キャッシュ・ミスが発生すると、プリミティブをレンダリングするために必要とされるテクスチャMIPマップ部分データが、ホスト・コンピュータ15の主メモリ17からキャッシュ・メモリ48へダウンロードされ、既に記憶されているなにがしかのデータを置き換えることになる。しかしながら、レンダリングされるプリミティブに関する一連のMIPマップ全体をダウンロードする従来技術のテクスチャ・マッピング・システムと異なって、本発明は、現時点でプリミティブをレンダリングする場合一連のMIPマップの実際に使用される特定部分または現時点でレンダリングされている部分のみをダウンロードする。詳細は後述するが、キャッシュ・ミスが発生すると、ホスト・コンピュータ15のテクスチャ割り込み管理機構を始動する割り込み制御信号が、テクスチャ・マッピング・チップ46によって生成される。割り込み制御信号は、ライン94を経由して分配器チップ30へ送られ、次に、ライン95を経由してホスト・コンピュータへ送られる。

【0030】要求されたテクスチャ・データが、ホスト・コンピュータによって主メモリから読み出され、3Dプリミティブ・レンダリング・パイプラインをバイパスして、バス24経由でテクスチャ・マッピング基板のメモリ48へダウンロードされる。このように、キャッシュ・ミス割り込みが発生する時、キャッシュ・ミスを引き起こしたプリミティブに関連するテクスチャ・データが主メモリ17からダウンロードされている間、フロントエンド基板が、3次元プリミティブに関する動作を継続し、テクスチャ・マッピング・チップおよびフレーム・バッファ基板へバス18を経由してプリミティブ・データを出力することができる。従来技術のテクスチャ・マッピング・システムと対照的に、テクスチャ・マッピング・ハードウェアへのテクスチャ・データのダウンロードが3次元プリミティブ・パイプラインのフラッシングを必要としないので、システムの帯域幅および処理能力が向上する。各ピクセルに関するテクスチャ・データは、テクスチャ・マッピング・チップ46によって、5本のバス28を経由してフレーム・バッファ基板へ送られる。5本のバス28は、フレーム・バッファ基板に備えられる5つのフレーム・バッファ制御器チップ50

A、50B、50C、50Dおよび50Eにそれぞれ接続され、それらフレーム・バッファ制御器チップに、計算結果のテクスチャ・データが並列的に送られる。フレーム・バッファ制御器チップ50A-50Eは、それぞれ対応するVRAM（ビデオ・ランダム・アクセス・メモリ）チップ51A-51Eグループに接続される。更にフレーム・バッファ基板は、4つのビデオ形式チップ（52A、52B、52Cおよび52D）およびRAMDAC（ランダム・アクセス・メモリ・デジタル・アナログ変換器）54を含む。フレーム・バッファ制御器チップは、表示画面の異なる非上重ねセグメントを制御する。各フレーム・バッファ制御器チップは、バス18経由でフロントエンド基板からプリミティブ・データを、そして、バス28経由でテクスチャ・マッピングから計算結果のテクスチャ・マッピング・データを受け取る。フレーム・バッファ制御器チップは、プリミティブ・データを補間して、それぞれの対応するセグメントに関する画面表示ピクセル座標、および各ピクセル座標に関して対応するオブジェクトR、G、Bカラー値を計算する。テクスチャ・マッピング基板から計算結果のテクスチャ・データが渡される（例えば三角形のような）プリミティブについて、フレーム・バッファ制御器チップは、ピクセル毎にオブジェクト・カラー値と計算結果のテクスチャ・データを結合して、表示画面上で表示されるべき最終的R、G、B値をピクセル毎に生成する。

【0031】オブジェクトとテクスチャ・カラー値の結合は、多くの異なる形態で制御することができる。例えば、置き換えモードでは、オブジェクト・カラー値が、単純にテクスチャ・カラー値によって置き換えられ、テクスチャ・カラー値だけがピクセルのレンダリングに使用される。別の形態の調整モードでは、オブジェクトとテクスチャ・カラー値が乗じられピクセルに関する最終的R、G、B値が生成される。更に、対応するテクスチャ・カラー値とオブジェクト・カラー値との組み合わせ方法を定める比率を指定するカラー制御ワードを各テクセルについて記憶することもできる。カラー制御ワードは、各ピクセルに対応するテクセル・データに関して決定され、フレーム・バッファ制御器チップにバス28経由で渡されるので、制御器チップは対応する制御ワードによって指定された比率を使用して最終的R、G、B値を各ピクセル毎に決定することができる。

【0032】フレーム・バッファ制御器チップ50A-50Eによって生成され、各ピクセルのR、G、B値を含む画像ビデオ・データが対応するVRAMチップ51A-51Eに記憶される。VRAMチップ51A-51Eの各グループは、40個のVRAMチップがフレーム・バッファ基板上に配置されるように、8つのVRAMチップを含む。ビデオ形式チップ52A-52Dの各々は、異なるセットの10個のVRAMチップに接続され、そこからデータを受け取る。ビデオ・データは、V

RAMチップから順次シフトされ、64ビット幅バス58A、58B、58Cおよび58Dを経由して4つのビデオ形式チップ52A、52B、52Cおよび52Dへ33MHz伝送率でそれぞれ送られる。ビデオ形式チップは、RAMDACが処理できるような形式にビデオ・データを変換して、形式化データを、32ビット幅バス60A、60B、60Cおよび60Dを経由して33MHz伝送率でRAMDAC54へ送る。次に、RAMDAC54は、デジタル・カラー・データをアナログR、G、Bカラー制御信号に変換し、各ピクセルに関するR、G、B制御信号を、R、G、B制御ライン29を介して表示画面に送る。

【0033】本発明の1つの実施形態において、特定のプリミティブ・レンダリング・タスクが複数のプリミティブに関して並列的に実行されるように、テクスチャ・マッピング基板12およびフレーム・バッファ基板14に関するハードウェアが反復配置され、これによって、システムの帯域幅が拡大される。そのような代替の実施形態の1例が図3に示されている。図3は、特定のハードウェアが複製されている本発明のコンピュータ・グラフィックス・システムのブロック図である。図3のシステムは、4つの3次元加速器チップ32A、32B、32Cおよび32D、キャッシュ・メモリ48Aならびに48Bとそれぞれ連動する2つのテクスチャ・マッピング・チップ46Aならびに46B、および、各々が対応するVRAMチップを持つ10個のフレーム・バッファ・チップ50A-50Jを含む。図3のシステムの動作は、図2のシステムのそれに類似している。図3の実施形態におけるハードウェアの反復配置によって、特定のプリミティブ・レンダリング・タスクが複数のプリミティブに関して並列的に実行されるためシステムの帯域幅が増大する。

【0034】II. テクスチャ・マッピング・チップの概要

図4は、テクスチャ・マッピング・チップ46のブロック図である。チップ46は、オブジェクトおよびテクスチャ・プリミティブ・データをフロントエンド基板から64ビット幅バス18経由で受け取るフロントエンド・パイプライン・インターフェース60を含む。テクスチャ・マッピング・チップ上で処理される三角形プリミティブは最高52個の32ビット・デジタル・ワードによって定義されるが、異なる長さのワードによって定義することもできる。パイプライン・インターフェースは、一組のマスター・レジスタと一組の対応するスレーブ・レジスタを含む。レンダリングの間、マスター・レジスタは、プリミティブを定義する52個のデジタル・ワード・データで逐次満たされる。次に、適切なレンダリング・コマンドを受領すると、データは、パイプライン・インターフェースのスレーブ・レジスタにシフトされ、これによって、マスター・レジスタはパイプライン方式

で別のプリミティブを表現するデータで満たされる。バス18經由で提供されるプリミティブ・データは、 x 、 y 、 z ベクトル座標データ、少なくとも1つの三角形頂点に関する S 、 T テクスチャ座標ならびに R 、 G 、 B オブジェクト・カラー・データ、および三角形平面方程式を表すデータを含む。上述のように、テクスチャ・マッピング・チップは、オブジェクト・ピクセル z 座標およびオブジェクト・カラー R 、 G 、 B 値を無視し、その他のデータだけをフロントエンド・パイプライン・インターフェース60に記憶する。

【0035】パイプライン・インターフェース60のスレーブ・レジスタは、バス62經由でパラメータ補間器回路64へプリミティブ・データを転送する。パラメータ補間器回路64は、各プリミティブ三角形を補間して、三角形を表現する各表示画面ピクセル座標について、ピクセルにマップする S 、 T テクスチャ・マップ座標、および、 S ならびに T 勾配値を決定する。 S ならびに T 勾配は、それぞれ、隣接するピクセルの間での S ならびに T 座標の変化に等しく、以下に説明される方法で計算される。

【0036】パラメータ補間回路64は、図5を用いて詳細は後述するが、辺ステッパ(edge stepper)66、FIFO(先入れ先出し)バッファ68、スパン・ステッパ(span stepper)70、勾配ならびに釣り合い補正回路72を全て逐次接続形態で含む。辺ステッパは、三角形頂点の1つの x 、 y ピクセル座標で開始し、三角形平面方程式を利用して、三角形の辺を辿って、三角形の辺を定義するピクセル座標を決定する。各ピクセル座標について、テクスチャ・マップにおけるどのテクセルが各表示画面ピクセル座標に対応するかを識別するように、三角形頂点の S 、 T 値に基づいてテクスチャ・マップの S ならびに T 座標が決定される。ピクセルおよびテクセル座標は、一時的にFIFOバッファに記憶され、次にスパン・ステッパに渡される。三角形の辺に沿った各 x 、 y ピクセル位置毎に、スパン・ステッパは三角形の対応する(x 、 y 間の)スパンに沿って進み、該スパンに沿ったピクセル位置のそれぞれについて S 、 T テクセル座標を決定する。

【0037】もしも、ピクセルが、テクスチャに関する一連のMIPマップの中の1つにおける単一のテクセルと1対1の対応関係を持たない場合、表示画面ピクセルに対する S ならびに T 座標の各々は、整数部分と小数部分を持つであろう。上述のように、テクスチャ・マップと対応付けされる時、各表示画面ピクセルが、テクスチャに関する一連のMIPマップの1つにおける複数のテクセルの間に落ちることがあり、更に、一連のMIPマップのサイズの隣接するMIPマップの間に落ちることもある。

【0038】勾配および釣り合い補正回路72は、各表示画面ピクセルに対する S および T の勾配値(ΔS およ

び ΔT)を決定する。本発明の1つの実施形態において、勾配 ΔS は、勾配 ΔS_x と勾配 ΔS_y のいずれか大なる方であるように選択される。ここで、勾配 ΔS_x は、表示画面上の隣接ピクセルの間で x 座標が変化することによって変わるテクスチャ・マップにおける S 座標の変化であり、勾配 ΔS_y は、表示画面上の隣接ピクセルの間で y 座標が変化することによって変わるテクスチャ・マップにおける S 座標の変化である。勾配 ΔT も同様に計算される。1つの表示画面ピクセルに関する ΔS および ΔT は、表示画面上のピクセルの対応する S 、 T 軸での変化に対するテクスチャ・マップ内の座標位置の変化率を示し、ピクセルに対するテクスチャ・データを作成するため、どのMIPマップがアクセスされなければならないかを決定するために使用される。例えば、表示画面ピクセルについて2に等しい勾配は、ピクセルが4つの(すなわち、後述されるように 2^2)のテクセルにマップすることを示して、当該ピクセルに関するテクスチャ・データを提供できるように基本マップからサイズの2だけ減じられたMIPマップ(例えば図1のマップ102)をアクセスしなければならない。かくして、勾配が増加することによって、ピクセルに対するテクスチャ・データを提供するためアクセスされるMIPマップのサイズは減少する。

【0039】本発明の1つの実施形態において、各ピクセルに対する適切なMIPマップを選択するため、勾配が当該ピクセルに関する ΔS_x 、 ΔS_y 、 ΔT_x および ΔT_y の最大値に等しくなるように、 ΔS および ΔT の大なる方に等しい単一の傾斜が使用される。しかし、勾配は、例えば上記の値の最小値、それら平均値あるいはその他の組合せを選択することによって、異なる形態で代替的に選択することも可能であることは理解されるべきであろう。 S 、 T 座標の1つだけの変化率を示す単一の勾配が選択されるので、その勾配の平方値は、対応するピクセルにマップするテクセルの数を表す。

【0040】勾配を使用して、パラメータ補間回路はピクセルが対応する最も近いマップと、ピクセルがそのマップに直接対応するものからどれほど離れているかを示す値を決定する。最も近いマップは、マップ番号の整数部分によって識別され、ピクセルがそのマップに直接対応するものからどれほど離れているかを示す値は、マップ番号の小数部によって識別される。

【0041】再び図4のテクスチャ・マッピング・チップのブロック図を参照して説明すれば、パラメータ補間回路64からのテクセル・データ出力が、ライン70經由でタイル作成器/境界検査器(tiler and boundary checker)72に送られ、そこで、テクセル・データによって指定されるテクスチャ・マップの各々の位置に最も近い4つのテクセルのアドレスが決定され、それらテクセルの各々がテクスチャ境界の内部にあるか否かが検査される。テクセル・データは、補間された S 、 T マップ座

標（整数値と小数値）、およびマップ番号ならびにマップ小数を含む。タイル作成器は、SおよびT座標の整数部分がパラメータ補間回路6によって計算された整数を使用し、各々の整数部に1を加えて4つの最も近いテクセルのアドレスを生成する。次に、境界検査器が、それら4つのテクセルのいずれかのS、T座標がテクスチャ・マップの境界の外側に落ちるか否かを判断する。もしもある表示画面ピクセルが、テクスチャ・マップの境界の外側に落ちるS、T座標位置に対応する場合、いくつかのマッピング・テクスチャ方式の1つによって、そのピクセルについてテクスチャ・データを生成すべきか、またそのデータをどのように生成すべきかが決定される。そのような方式の例には、ラッピング（wrapping、すなわちテクスチャの繰り返し）、ミラーリング（mirroring、すなわちテクスチャの鏡画像の繰り返し）、境界の外側にあるテクスチャ・マッピングの取り消し、および、境界外での同一調カラー表示などが含まれる。

【0042】境界を越えたテクスチャ・マップ位置にピクセルをマップすることを可能にすることによって、テクスチャをオブジェクト・プリミティブにマップする方法に柔軟性が与えられる。例えば、テクスチャがオブジェクトの複数部分にマップされるように、反復動作でテクスチャをオブジェクトにマップすることが望ましい場合がある。例えば、[0, 0]から(10, 10)までの範囲のS、T座標を持つテクスチャが定義される場合、ユーザは、そのような範囲のS、T座標へマップするようにオブジェクトの特定部分を指定することができる。上記において、記号[]は、指定する範囲が括弧内座標を含み、記号()は括弧内座標を含まないことを表し、以下においても同様の表記法を使用する。ラッピング機構がテクスチャの境界の外側に落ちるS、T座標について動作するように選択される場合、[10, 10]から(20, 20)までのS、T座標を持つピクセルは、[10, 10]から(20, 20)までのS、T座標にあるテクセルにそれぞれ対応するであろう。

【0043】上述のように、1つのピクセルに関して2次元テクスチャ・マップから得られるテクスチャ・データは、8つのテクセル、すなわち最も近い2つのMIPマップにおける最も近い4つのテクセルが結合された結果である。8つのテクセルを結合してテクセル・データを生成する多数の方法がある。例えば最も近いマップにおける最も近い単一のテクセルを選択することによって、平均算出の必要性をなくすることができる。別の方法として、最も近い2つのマップそれぞれの最も近い単一のテクセルが、勾配値に基づいて平均される。このような方法は、8つの最も近いテクセルの平均値が計算される場合のように正確にテクスチャをマップしない。

【0044】本発明の1つの実施形態においては、単一ピクセルに関するテクスチャ・データを8つのテクセルの加重平均として計算する3線形補間法(trilinear int

erpolation)が利用される。テクスチャ・データをアクセスする最も近い2つのMIPマップを識別するためS、Tの変化率を表す勾配が使用され、各々のマップ内の最も近い4つのテクセルがアクセスされる。表示画面ピクセルがマップするMIPマップの位置のS、T座標に最も近いテクセルに基づいて、各マップ内の4つのテクセルの平均が加重される。該ピクセルに関するS、T座標の小数部分が、この加重を実行するために使用される。次に、最も近い2つのMIPマップ各々の上記平均値が、勾配値に基づいて加重される。この加重プロセスにおける使用のため、勾配を基に小数値が計算される。例えば、値3の勾配は、勾配2および勾配4にそれぞれ対応するMIPマップの中間にある。

【0045】テクセル補間プロセスは、テクセル補間回路76によって実行される。各表示画面ピクセルに関するSおよびT座標の小数部分は、タイル作成/境界検査器を経由して、パラメータ補間回路からテクセル補間回路76へライン74を介して送られる。テクセル補間回路は小数部分を使用して、複数のテクセルの各々に与えられる加重を決定し、所望のテクセル・データを計算する。

【0046】上述のように、レンダリングされるプリミティブに関連するテクスチャMIPマップは、ローカル・キャッシュ・メモリ48（図2）に記憶される。本発明の1つの実施形態において、キャッシュは完全連想型である。キャッシュは、各インターリーブに2つのSDRAMチップが配置される構成で、全体として4つのインターリーブに区分けされる8つのSDRAMチップを含む。各インターリーブ内のSDRAMチップが同時にアクセスされるように各インターリーブに対応して1つ宛計4つのコントローラが備えられる。各SDRAMチップは2つのメモリ・バンクを含む。上記メモリ・バンクにおいては、従来技術のDRAMの場合に起きるような2つの異なるページ（すなわち2つの異なる行アドレス）からデータを取り出すことに一般に関連する再ページングの負荷を伴うことなく、メモリの異なるページを連続的読取りサイクルでアクセスすることができる。

【0047】テクスチャ・データ（すなわち、MIPマップ）は、各々が256×256のテクセルを含むテクセル・データ・ブロックに分割される。キャッシュ・メモリは、一時点で64個のデータ・ブロックを記憶することができる。各ブロックは、ブロックをユニークに識別するブロック・タグを持つ。キャッシュは、キャッシュに現在記憶されているデータ・ブロックに対応するブロック・タグを記憶するディレトリ78を含む。詳細は後述するが、ブロック・タグの各々は、データ・ブロックが表現する特定のテクスチャを識別するテクスチャ識別子（すなわちテクスチャID）、当該テクスチャの一連のマップの中からデータ・ブロックが表す特定のMIPマップを識別するマップ番号、および、該特定マッ

ブ内の上記データ・ブロックの位置を識別する高位SならびにT座標を含む。キャッシュ・ディレクトリ内のブロック・タグの物理的位置が、キャッシュ・メモリ内における対応するデータ・ブロックの位置を表す。

【0048】異なるテクスチャを区別するテクスチャ識別子を用いて、複数のテクスチャのMIPマップをキャッシュ・メモリに同時に記憶することもできる。一部のMIPマップが256×256未満のテクセルを含むこともあり、この場合データ・ブロックの一部は使用されない。例えば、一連のMIPマップの小さい方のマップ、または小さいテクスチャの場合大きい方のマップでも、256×256個のテクセルを越えないことがある。メモリ空間を有効に活用するため、各マップ部分がブロック内のサブブロックに割り当てられるように、複数のマップ部分がテクスチャ・データの1つのブロック内に記憶されるようにすることもできる。1つのブロック内に記憶される複数のマップの各々は、ブロック内のマップの位置を識別するサブテクスチャ識別子(ID)を持つ。

【0049】レンダリングの間、タイル作成/境界検査器72は、レンダリングされるピクセルに対応するテクスチャ・データ・ブロックに関する読取りキャッシュ・タグを生成する。タグを生成する方法の詳細は後述する。タグは、テクスチャ・データのテクスチャIDを表す8ビット、テクスチャ・データのマップ番号を決定する際に使用される1ビット、および、テクスチャ・データの高位7ビットのSならびにT座標を含む23ビットのフィールドである。キャッシュ・ディレクトリ78は、タイル作成/境界検査器から送られる読取りキャッシュ・タグをディレクトリに記憶されているブロック・タグと比較して、レンダリングの際に使用されるべきテクスチャ・データ・ブロックがキャッシュ・メモリに存在するか否かを判断する。レンダリングされるべきプリミティブに対応するテクスチャ・データ・ブロックがキャッシュ・メモリに記憶されている場合(すなわちキャッシュ・ヒットの場合)、キャッシュ・ディレクトリは、ヒットしたタグに対応するキャッシュ内のテクスチャ・データ・ブロックの物理的位置を標示するブロック・インデックスを生成する。ブロック・インデックスの計算の詳細は後述する。キャッシュから読み取られるべき各テクセルについて、ブロック内のテクセルの位置を標示するテクセル・アドレスがまたタイル作成/境界検査器72によって生成される。テクセル・アドレスは、より大きいサイズ・マップに関する補間されたS、T座標の低位アドレス・ビットを含み、より小さいサイズのマップに関して以下に記述されるアルゴリズムに基づいて計算される。ブロック・インデックスおよびテクセル・アドレスとともに、キャッシュ内のテクセルの位置を示すキャッシュ・アドレスを含む。詳細は後述するが、4つのインターリーブのどこにテクセルが記憶されてい

るかを決定するため、各テクセルに関するSならびにT座標のLSB(すなわち最下位ビット)がデコードされ、キャッシュ・アドレスの残りのビットは、コマンドと共に、ライン84経由でテクセル・キャッシュ・アクセス回路82へ送られ、キャッシュ内の上記アドレス位置に記憶されているテクセル・データが読み取られる。

【0050】読取りキャッシュ・タグがキャッシュ・ディレクトリに記憶されてるブロック・タグのいずれとも一致しない場合、すなわちキャッシュ・ミスが発生する場合、キャッシュ・ディレクトリ78は、ライン94

(図2) 経由でフロントエンド基板上に対して割込み制御信号を生成し、これに回答して、分配器チップ30がライン95経由でホスト・コンピュータ15に対する割込みを生成する。割込みに回答して、ホスト・コンピュータのプロセッサ19が、サービス・ルーチン(詳細は後述)を実行することによって、キャッシュ・ミスのあったブロック・タグをキャッシュ・ディレクトリから読み取り、フロントエンド基板10およびテクスチャ・マッピング・チップ46における3次元プリミティブ・パイプラインをバイパスする形態で、テクスチャ・データの対応するブロックをキャッシュ・メモリにダウンロードする。主メモリからダウンロードされたテクスチャ・データは、バス24経由で(図4の)テクセル・ポート92を通してテクセル・キャッシュ・アクセス回路82へ送られ、キャッシュ・メモリを形成するSDRAMへ書き込まれる。

【0051】キャッシュ・ミスが発生する時、テクスチャ・マッピング・チップは、ミスが発生したプリミティブの処理を進める前に、新しいテクスチャ・データがダウンロードされるのを待つ。しかしながら、キャッシュ読み取りに続くパイプラインの処理段階は、ミスのあったプリミティブに先行して受け取ったプリミティブを処理し続ける。同様に、キャッシュ読み取りに先行するパイプラインの処理段階は、新しいテクスチャ・データのダウンロードを待っている間、パイプラインがいつぱいにならない限り、キャッシュ読み取り動作の背後でプリミティブの処理を続行する。

【0052】レンダリングの間、フレーム・バッファ基板14におけるパイプラインの後の方の処理段階は、対応するテクスチャ・データがテクスチャ・マッピング基板から受け取られるまで、プリミティブの処理を進めない。従って、キャッシュ・ミスが発生して、テクスチャ・マッピング・チップが新しいテクスチャ・データのダウンロードを待つ時、フレーム・バッファ基板14は、同様に、テクスチャ・マッピング・チップから送られてくるテクスチャ・データを待つ。テクスチャ・マッピング・チップの場合と同様に、テクスチャ・マッピング・データの受け取り段階に続くパイプラインの処理段階は、キャッシュ・ミスのあったプリミティブに先立って受け取ったプリミティブの処理を続行し、テクスチャ・

マッピング・データを受け取る段階に先行するパイプラインの処理段階はパイプラインがいっぱいにならない限りプリミティブの処理を続行する。

【0053】キャッシュ・ミスにตอบสนองして新しいテクスチャ・データを待つ時テクスチャ・マッピング基板またはフレーム・バッファ基板いずれかのパイプラインが待機する場合、フロントエンド基板10のパイプラインもまた同様に待機するする点は理解されるべきであろう。キャッシュ・ミスの発生によって、ホスト・コンピュータの主メモリへのアクセスおよびテクスチャ・データのダウンロードを完了するにはいくつかのサイクルがかかるので、フレーム・バッファ基板のパイプラインが待機させられたことによってテクスチャ・マッピング・チップのパイプラインが待機する必要がないことを確認することが望ましい。従って、本発明の1つの実施形態においては、フレーム・バッファ基板が、テクスチャ・マッピング基板より深いプリミティブ・パイプラインを備えるように構成され、それにより、フレーム・バッファ・パイプラインが使用可能になるのを待つことによるテクスチャ・マッピング・パイプラインの遅延がなくなる。

【0054】本発明の1つの実施形態では、上記の機能を備えさせるため、テクスチャ・マッピング機能がオフにされる。これは、ホスト・コンピュータのプロセッサ19上でソフトウェアを操作して、テクスチャ・マッピング基板12およびフレーム・バッファ基板におけるレジスタを設定することによって達成される。テクスチャ・マッピングがオフに設定される時、これらのレジスタはそれぞれ、テクスチャ・マッピング・チップ46がフレーム・バッファ基板14へテクスチャ・データを送ることを禁止し、テクスチャ・マッピング基板からのテクスチャ・データを待つことなくプリミティブに対するレンダリングを続けるようにフレーム・バッファ基板に命令する。

【0055】上述のように、2次元テクスチャ・マップからのテクスチャ・データでレンダリングされる表示画面ピクセルの各々について、(双線形補間の場合)1つのMIPマップから4つのテクセル、または(3線形補間の場合)2つの隣接MIPマップから8つのテクセルが、キャッシュ・メモリから取り出され、該ピクセルに対するテクスチャ・データが決定される。キャッシュから読まれたテクセルは(図3の)バス86経由でテクセル補間回路76へ送られ、そこで、複数テクセルの補間によって、各ピクセルのテクセル・データが計算される。補間方法は、システムに関して設定されるモードに応じて変り得る。1点標本抽出補間モードが設定される場合、結果として生成されるテクセル・データは、テクスチャ・マップにおけるピクセルのS、T座標によって定義される位置に最も近い1つのテクセルに等しい。別の方法として、双線形補間または3線形補間が用いられる場合、それぞれ1つまたは最も近い2つのマップにお

ける4または8個の最も近いテクセルの加重平均である。複数のテクセルの各々に与えられる加重は、タイル作成/境界検査器からテクセル補間回路76へ提供される勾配値およびSならびにT座標の小数部分に基づいて決定される。

【0056】表示画面ピクセルに関する計算結果のテクセル・データは、バス88経由でフレーム・バッファ・インターフェースFIFOバッファ90へ順次送られる。フレーム・バッファ・インターフェースFIFOバッファ90は、最高64までの計算結果のテクセルを記憶することができる。

【0057】計算結果のテクセルの各々は、R、G、Bを表現する各8ビット、および α を表す8ビットを含む32ビット・ワードである。 α バイトは、(図2の)フレーム・バッファ基板14に対して、テクセルに対応するピクセルについて最終的表示画面R、G、B値を計算する際に、計算結果のテクスチャ・データのR、G、B値をフレーム・バッファ基板によって生成されたオブジェクト・データのR、G、B値と結合する方法を標示する。フレーム・バッファ・インターフェースFIFOバッファ出力T0-T4は、(図2の)バス28を経由してフレーム・バッファ基板14へ送られる。フレーム・バッファ基板は、各画面表示ピクセルについて最終的R、G、B値を生成するため α によって指定された方法で、計算結果のテクセル・データのR、G、B値をオブジェクトR、G、値と結合する。

【0058】III. キャッシュ・メモリの構成

図6は、本発明の1つの実施形態に従うキャッシュ・メモリのブロック図である。該キャッシュ・メモリは、テクセル・ポート92、テクスチャ補間回路76、キャッシュ・ディレクトリ78およびテクセル・キャッシュ・アクセス回路82を含むテクスチャ・マッピング・チップ部分に接続する。この実施形態では、キャッシュ・メモリ48は、4つのインターリーブ204A、204B、204Cおよび204Dを含む。各インターリーブは、同時にアクセスされることができる2つのSDRAMチップ(図示されていない)を含む。各SDRAMは1回の読み取りサイクルの間に8ビットのデータを提供する。従って、各インターリーブは、1読み取りサイクルの間に16ビットのテクセル・データを提供する。インターリーブの各SDRAMの2つの連続する位置のそれぞれに8ビットが記憶される形態で、1つのインターリーブのキャッシュに各々32ビット・ワードのテクセル・データが記憶される。従って、キャッシュから1つのテクセルを読み取るためには、該当するインターリーブの連続的位置に対する2回の読み取りサイクルの実行によって32ビットのテクセル・データが取り出される。後述されるように、2回の連続サイクルでバス・データ(burst data)を作成するには、(行および桁データを含む)1つのアドレス・ワードだけを各インターリーブ内

のSDRAMに送出すればよい。バースト・データは、与えられたアドレスから第1のサイクルで渡される16ビット、同じ行を持つアドレスから第2のサイクルで渡される16ビット、および1だけ増分される桁を含む。

【0059】テクセル・キャッシュ・アクセス回路82は、コントローラA(200A)、コントローラB(200B)、コントローラC(200C)およびコントローラD(200D)と名付けられた4つの独立したコントローラを含む。4つのコントローラA、B、CおよびDは、並列バス202A、202B、202Cおよび202Dを経由して4つのインターリーブ204A、204B、204Cおよび204Dのデータに同時にアクセスすることができる。上記コントローラは、バス84A、84B、84Cおよび84D経由でそれぞれ受け取ったコマンドおよびアドレスに应答してキャッシュ・メモリ48からテクセル・データを読み取る。

【0060】上述のように、各ピクセルは、潜在的に、1つのMIPマップの4つのテクセルに対応するか、あるいは複数のMIPマップの8つのテクセルに対応する。詳細は後述するが、キャッシュにダウンロードされるテクセル・データは、ホスト・コンピュータの主メモリにおいて、各MIPマップにおけるいかなる4つの隣接するテクセルも並列的アクセスが可能のように別々のインターリーブに位置づけられるように配置される。従って、双線形補間法によってテクセル・データを生成するために必要とされるMIPマップにおけるいかなる4つの隣接するテクセルも、1回の読取り動作で読み取ることができる。3線形補間法が使用される場合は、1組4つからなる2組の(計8つの)テクセルが隣接するMIPマップから2回の読取り動作で読み取られる。

【0061】図7は、キャッシュ・メモリの4つのインターリーブ構成によってある1つのMIPマップにおける任意の4つの隣接テクセルを同時に読み取ることができる利点を活かすようにテクスチャ・データ・ブロックが配置される形態の1例を示している。各テクセルには、該テクセルが記憶されているキャッシュ・メモリのインターリーブを識別するラベルA、B、CおよびDが付けられている。マップ内のいかなる位置もA、B、CおよびDのラベルを持つ4つのテクセルの間に落ちるように、AないしDのラベルのパターンが繰り返されている。このようにして、あるマップの内の任意の位置に対応するピクセルについて、最も近い4つのテクセルは、別々のインターリーブAないしDに存在するため、それらテクセルは4つの独立コントローラ200A-200Dによって同時にアクセスされることができる。例えば、ピクセルP0はA、B、CおよびDというラベルの4つのテクセルの間の位置に対応し、ピクセルP1はB、A、DおよびCというラベルの4つのテクセルの間の位置に対応する。

【0062】上述のキャッシュ構成は例示の目的で記述

したものであって他の代替構成も実施できることは理解されるべきであろう。例えば、キャッシュを、8つの別々のコントローラを持つ8つの別々のインターリーブの形態で実施して、3線形補間法が使われ時、8つのテクセルが、1回の読取り動作で同時にアクセスされることができるように構成することもできる。

【0063】キャッシュ・メモリ内のSDRAMチップの各々は、同時に別々の活動ページ(すなわち、共通の行アドレスを持つメモリ位置グループ)を維持することができる2つの等しいサイズのバンクに内部的に分割される。このようにして、従来技術のDRAMの場合に起きるような2つの異なるページ(すなわち2つの異なる行アドレス)からデータを取り出すことに一般に関連する再ページングの負荷を伴うことなく、SDRAMチップの2つのバンク内の異なるページにあるデータを連続的に読取りサイクルでアクセスすることができる。

【0064】更に詳細は後述するが、3線形補間法を使用する際のページ間アクセス負荷を最小限にとどめるこのSDRAM構成の利点を活かすようにテクスチャ・データがキャッシュ・メモリ内に配置される。3線形補間のために必要な8つのテクセルは、2つのMIPマップに収納されている1組4テクセルからなる2組のテクセルを含む。1つのMAPにある1組の4つの隣接するテクセルの各々は、上述のように、同時にアクセスできるようにインターリーブA、B、CおよびDにそれぞれ記憶されている。更に、任意のテクスチャについて一連のMIPマップにおける隣接するMIPマップ上で共通するデータが、キャッシュの異なるSDRAMバンクに記憶される。3線形補間が実行される時、第1のバーストの2回の読取りサイクルの間に、1つのMIPマップの4つのテクセルがインターリーブAないしDのSDRAMバンクの1つから同時に読み取られ、後続のバーストの2回の読取りサイクルの間に、隣接するMIPマップの4つのテクセルが別のSDRAMバンクから読み取られる。SDRAMの両方のバンクは同時に行が有効であるので、再ページングの負荷なしに2組のテクセルは連続的なバースト・モードで読み取られる。オブジェクトの複数ピクセルがレンダリングされる場合、隣接ピクセルは、該テクスチャに関する同じ2つのMIPマップに対応していることが多いため、キャッシュへの読取りが、2つのマップに共通データを記憶するキャッシュ・ブロックの間に連続的に切り換えを行うことが必要とされる点は理解されるべきであろう。表示画面ピクセルをレンダリングする間2つの隣接MIPマップの間での切り換えを行う時、各サイクルの再ページングの負荷を伴うことなく3線形補間を実行できるので、2つのページが各SDRAM内で活動的であることを可能にする本発明のキャッシュ構成は利点がある。

【0065】図8は、本発明のキャッシュ・メモリの上述の実施形態のさらに詳細なブロック図である。キャッ

シュは、各々が2つのSDRAMチップを含む4つのインターリーブ204A-204Dに均一に分割されたSD1-SD8というラベルの8つのSDRAMチップを含む。各インターリーブの2つのSDRAMは、以下のような共通のラインを共有する。すなわち、11本のアドレス線(ADD)、行・桁ストロブ(RAS並びにCAS)、書き込みイネーブル(WE)、クロック・イネーブル(CKE)、およびデータ入出力マスク(DQM)である。各インターリーブ内のSDRAMは、各読取りまたは書き込みサイクルの間8ビットデータがそれぞれ読み書きされる8本の独立データ線に接続される。各SDRAMチップは、各々がテクスチャ・データの1,048,576個の8ビット・ワードを記憶する2つのメモリ・バンクを含む。

【0066】各インターリーブの2つのSDRAMは同時にアクセスされ、一方のSDRAMがデータ・ビット[15:08]を、他方がデータ・ビット[07:00]をそれぞれ提供することによって両方で16ビットのデータを提供する。上述のように、1回のバースト・モードの2つの連続的読取りサイクルが、各インターリーブから32ビット・テクセル・データを読み取る。その個々の8ビット・ワードは、当該テクセルR, G, Bおよびα値の各々を表す。

【0067】SDRAMチップが、11本のアドレス線ADD上で多重送信された20のアドレス・ビットを受け取り、各バンク内の1,048,576個の8ビット・ワードをデコードする。詳細は後述するが、キャッシュからアクセスされるべき各テクセルについて6ビットのブロック・インデックスおよび16ビットのテクセル・アドレスが計算される。ブロック・インデックスは、64個のデータ・ブロックのどこにテクセルが位置しているか標示し、テクセル・アドレスは、ブロック内のテクセルの正確なS, T座標アドレスを標示する。1平方データ・ブロックが256×256テクセルを含むと仮定すれば、8つのSビットおよび8つのTビットがテクセル・アドレスを構成する。キャッシュ・アドレスは、ブロック・インデックス(MSB 6ビット)およびテクセル・アドレス(LSB 16ビット)の組合せを含む22ビット・ワードである。キャッシュ・アドレスは、キャッシュ内の正確なテクセル位置を示す。

【0068】レンダリングの間、タイル作成/境界検査器が、テクセル・アドレスの下位Sビットおよび下位Tビット(すなわち、LSB S座標およびLSB T座標)をデコードして、テクセルがキャッシュの4つのインターリーブのどれに記憶されているかを決定する。キャッシュ・アドレスの残りの20アドレス・ビットは、アドレス線ADDに沿って、該当するインターリーブ内の2つのSDRAMチップに対して提供される。2つのSDRAMチップに対して提供される20アドレス・ビットのうち、9ビットは、テクセル・データにアクセス

するためSDRAM内の桁を選択するために使用され、11ビットは行を選択するために使用される。当業者によって理解されるように、桁および行アドレス・ビットは、異なるサイクルでSDRAMにラッチされ、RASおよびCASストロブは、従来技術の方法でデータにアクセスするために使用される。

【0069】2サイクルのバースト・モードの間に、第1のサイクルの間に同じインターリーブ内の2つのSDRAMのアドレス指定された位置から16ビットが読み取られ、次に、別のアドレスを用意することなく、第2のサイクルで、2つのSDRAMのもう1つ別の位置から16ビットが読み取られる。第2のサイクル中のアドレスは、同じ行アドレスおよび1増分された桁アドレスを含む。一旦1つのページ(すなわち特定の行)のアドレスが起動されれば、異なる行アドレスが与えられるまでその行は活動的であるという点は理解されるべきであろう。従って、同じインターリーブからアクセスされるべき連続的テクセルが(同じ行アドレスを含む)同じページにあるならば、行アドレスは、連続バーストの最初に1度だけ提供されればよい。

【0070】加えて、RAS、CASおよびWEラインは、従来技術の方法でアドレス指定しSDRAMチップへデータを書き込むため使用される。クロック・イネーブル信号CKEがオフにされると、内部クロックは中断される。SDRAMは、この信号に応答してデータを処理しない状態に保ち、両方のバンクをアイドル状態にする。データ入出力マスクDQM信号は、読取りサイクルの間出力イネーブルとして機能し、書き込みサイクルの間入力データ・マスクとして機能する。

【0071】従来技術におけるSDRAMの使用方法では、SDRAMは、現在ページから現在のデータをアクセスしている間に後続のデータをどのページからアクセスするかを決定して現在データ読取りサイクルが完了する前にその将来のページを起動させる。SDRAMが、2つの異なるページをイネーブルして同時に活動的にさせるので、上記従来技術のSDRAMの使用は、従来技術のDRAMの使用におけるようなデータを異なるページからアクセスする場合に派生する再ページングの負荷を回避する。しかしながら、多数の連続読取りサイクルで読み取られるべきデータが異なるページに位置する場合、将来ページを前もって調べ起動するために複数のサイクルが必要とされるため、従来技術のSDRAM使用法は上記の利点を提供しない。本発明のテクスチャ・データ記憶方法は、再ページングなしで異なるページからの複数の連続的SDRAM読取りサイクルを実行可能とさせることによって、従来技術のSDRAM使用に比較して利点を持つ。特に、(3線形補間を実行する場合連続的読取りサイクルの間のアクセスを必要とする)テクスチャの隣接MIPの共通データをSDRAMの別々のバンクに記憶することによって、別々のバンクから

のデータが、連続的な読取りサイクルで再ページングの負荷なしにアクセスされることができる。SDRAM処理性能を向上させるための本発明のデータ記憶配置方法をテクスチャ・マッピング・データに関して以上説明したが、本発明の方法がそのようなテクスチャ・マッピング・データに関するものに限定されない点は理解されるべきであろう。特に、複数の連続的な読取りサイクルが異なるメモリ位置からデータをアクセスするようなタイプのデータのすべてを割り当てる場合に本発明の方法は応用できる。

【0072】IV. キャッシュ制御FIFO

図9は、境界検査器72、キャッシュ・ディレクトリ78、キャッシュ・アクセス回路82、キャッシュ・メモリ48およびテクセル補間回路76を含むテクスチャ・マッピング・チップの一層詳細なブロック図である。テクセル・キャッシュ・アクセス回路82は、4つのキャッシュ・アクセス・コマンドFIFO206A、206B、206Cおよび206Dを含む。キャッシュ・アクセス・コマンドFIFO206A-206Dは、16ビット・バス84A、84B、84Cおよび84Dを経由して境界検査器からそれぞれ受け取るキャッシュ・アクセス・コマンドを記憶する。キャッシュ・アクセス・コマンドFIFO206A-206Dは、図6に示されるコントローラ200A-200Dにそれぞれ対応する。例えば、FIFO206Aコマンドは、インターリーブ204A内のSDRAMのキャッシュ・アクセスを起動する。この実施形態においては、各キャッシュ・アクセス・コマンドFIFOは、8つの16ビット・コマンドを一時的に記憶する能力を持つ。このように、システムのパイプライン性能を向上させるため、キャッシュ・アクセス回路が働く前に、8つのコマンドがキャッシュ・アクセス・コマンドFIFOの各々に記憶される。

【0073】上述のように、レンダリングの間、境界検査器72は、対象のピクセルに対応するテクスチャ・データのブロックの各々に関する読取りキャッシュ・タグをキャッシュ・ディレクトリ78に記憶されているブロック・タグの各々と比較して、テクセルがキャッシュにあるかどうかを判定する。ヒットが発生すれば、キャッシュ内のテクスチャ・データの対応するブロックの位置を表すブロック・インデックスが生成される。タイル作成/境界検査器は、補間S、T座標、テクスチャID、特定テクセルのサブテクスチャID、テクセルをアクセスすべきマップのマップ番号およびテクスチャの基本マップのサイズを用いてテクセル・アドレスを決定するルーチンを同時に実行する。詳細は後述する。(キャッシュ・アドレスを構成する)ブロック・インデックスおよびテクセル・アドレスを用いて、タイル作成/境界検査器がテクセルが記憶されているキャッシュの特定のインターリーブおよびそのインターリーブのSDRAMチップの行および桁アドレス・ビットを決定する。アドレス

情報は、キャッシュ読取りコマンドとともに、対応するキャッシュ・アクセス・コマンドFIFOに送られる。

【0074】テクセル補間回路76は、8つのテクセル・データFIFO214A0、214A1、214B0、214B1、214C0、214C1、214D0および214D1を含む。テクセル・データFIFO214A0ならびに214A1は、キャッシュ・メモリのインターリーブ204Aに対応し、FIFO214B0ならびに214B1は、インターリーブ204Bに対応し、FIFO214C0ならびに214C1は、インターリーブ204Cに対応し、FIFO214D0ならびに214D1は、インターリーブ204Dに対応する。

【0075】先に述べたように、キャッシュ・メモリの4つのインターリーブの各々は、別々のキャッシュ・アクセス経路を通して同時にアクセスされることができる。レンダリングの間、テクセル・キャッシュ・アクセス回路82がキャッシュ・メモリ48からテクセル・データをアクセスする時、テクセル・アクセス制御ワードが、バス208A、208B、208Cおよび208Dを経由してキャッシュ・メモリ48へ与えられる。2つの連続する16ビットの読取りサイクルの間に4つのテクセルが同時に4つのインターリーブからアクセスされる。4つのテクセルは、バス210A、210B、210Cおよび210D経由で、テクセル・データAのFIFOの1つ(214A0または214A1)に、テクセル・データBのFIFOの1つ(214B0または214B1)に、テクセル・データCのFIFOの1つ(214C0または214C1)に、テクセル・データDのFIFOの1つ(214D0または214D1)にそれぞれ送られる。各インターリーブA-Dに対応するFIFOのペア(すなわち0および1)は交互にロードされる。例えば、インターリーブAから読み取られる第1のテクセルがテクセル・データFIFO214A0に記憶され、インターリーブAから読み取られる第2のテクセルがテクセル・データFIFO214A1に記憶され、インターリーブAからの第3のテクセルがテクセル・データFIFO214A0に記憶されるというようにテクセルが交互に記憶される。このような交互方式を使用する理由は以下の通りである。

【0076】テクセル・データFIFOの各々は、幅32ビットで深さ8段階である。組合せれば、8つのFIFO214は、8つのパイプライン化された段階を記憶する。各段階は、3線形補間の間所望のテクセル・データを決定するために使用される8つのテクセルを含む。バス210A、210B、210Cおよび210Dは、幅16ビットである。各インターリーブにおける各SDRAMペアは、各読取りサイクルの間に16ビットのデータを提供する。各バースト読み取りの間、第1の16ビットが、各SDRAMペアから第1の16ビット・レ

ジスタ（図示されていない）に送られ、次の16ビットが、各SDRAMペアから第2の16ビット・レジスタ（やはり図示されていない）に送られる。バースト読み取りの第2サイクルの終了時点で、両方のレジスタからのデータが、対応する32ビット・バス212A、212B、212Cまたは212D上へ送られる。任意のピクセルに関する所望のテクセル・データを決定するため、テクセル補間回路76が、FIFOにアクセスして次の段階の8つのテクセルを読み取り、上述の方法でそれらのテクセルを補間する。補間結果のテクセル・データが、バス28経由でフレーム・バッファ基板14

（図2）へ送られ、そこで、上述の方法で表示画面ピクセルをレンダリングするために使用される。

【0077】3線形補間が実行される時、任意のピクセルに関する所望のテクセル・データは、ある1つのMIPマップの4つのテクセルを補間し、隣接する別のMIPマップの4つのテクセルを補間して得られる。隣接する表示画面ピクセルは、一般的には連続的にレンダリングされる。隣接する表示画面ピクセルは、頻繁に、1つのテクスチャMIPマップの隣接位置に対応する。この結果、連続的にレンダリングされるプリミティブに関して所望のテクセル・データを補間する際いくつかの共通のテクセル・データが使用されることがよくある。本発明の1つの実施形態において、多数の近接した読み取りサイクル内で共通のテクセル・データが多数回アクセスされる場合、キャッシュは、最初の読取りについてのみアクセスされるだけで、後続の読取りの各々についてはキャッシュ読取りサイクルを節約する。最も最近読まれたテクセルが、テクセル・データFIFO内に記憶される。このように、それらテクセルに対する後続のアクセスは、キャッシュではなくFIFOからなされる。これによって、必要とされるキャッシュ・アクセス数が減り、システムの帯域幅が増大する。

【0078】テクセル・データ経路A、B、CおよびDの各々について、前回のピクセルに関してテクセル・データFIFO0または1の1つに最も最近書き込まれたテクセル・データが、キャッシュをアクセスするためパイプライン位置に現在あるテクセル・データと一致する場合、キャッシュ・アクセス・コマンドは、対応するキャッシュ・アクセスFIFO206A、B、CまたはDに送られない。代わりに、テクセル・データが対応するテクセル・データFIFO214A、B、CまたはDの最も最近書かれた位置に記憶されていることを示すコマンドがテクセル補間器に送られる。キャッシュをアクセスするためパイプライン位置に現在あるテクセル・データに対応するテクセル・データが、対応するテクセル・データFIFOの最も最近書き込まれた位置のデータと一致しない場合、経路A、B、CおよびDのいずれについても、テクセル・キャッシュ・アクセス・コマンドが対応するテクセル・キャッシュ・アクセス・コマンドF

FIFOに送られ、キャッシュ・メモリ48からそのテクセル・データが読み取られる。

【0079】キャッシュ・アクセスを考慮しなければならない現在パイプライン位置にあるピクセルについてインターリーブAないしDのいくつか異なる結果を生み出す点は理解されるべきであろう。例えば、連続的ピクセルに関する共通のテクセル・データがインターリーブAには存在するがインターリーブB-Dには存在しないことがある。そのような状況においては、キャッシュからテクセル・データをアクセスするためにパイプライン位置にある第2の連続ピクセルに関してテクセル・データがインターリーブB-Dから読み取られるであろうが、その第2のピクセルに関するインターリーブAからのテクセル・データは、テクセル・データFIFO214A0または214A1の1つの同じ位置から読み取られるであろう。キャッシュをアクセスせずに複数のピクセルに関してテクセル・データFIFOからテクセルが再読み取りされる場合、本方式は帯域幅を節約する。

【0080】テクセル補間回路76は、53ビット・コマンドを境界検査器72から53ビット・バス218経由で受け取るテクセル補間回路コマンドFIFO216を含む。テクセル補間回路コマンドFIFOは、各サイクルの間所望のテクセル・データを補間する際に使用されるべきテクセル・データをどのテクセル・データFIFO位置が含むかを補間回路に標示する最高16のコマンドを記憶することができる。補間回路コマンドは、また、（点標本抽出、双線形または3線形などの）補間モードを示し、補間の際に各テクセルが加重される方法を指定するSおよびT座標の勾配および小数値を含む。コマンドは、（双線形補間の場合）4個または（3線形補間の場合）8個のテクセルがFIFO214A0、A1、B0、B1、C0、C1、D0またはD1のいずれから読み取られるべきか、また、該テクセルが新しいか古いかを示すデータを含む。テクセル・データがその経路のいずれかのテクセル・データFIFOの位置に最も最近書き込まれたテクセル・データと異なる場合は、そのテクセル・データは新しいという。新しい場合キャッシュ読取りが必要とされる。テクセル・データがいずれかのテクセル・データFIFOの位置に最も最近書き込まれたものと同じ場合は、そのテクセル・データは古いという。古い場合キャッシュ読取りは必要とされない。テクセル・データが新しい時、FIFO読取りポインタが、FIFOの内の次の位置へ移動されなければならない。一方、テクセル・データが古い時、同じデータが同じFIFO位置から読み取られ、読取りポインタを移動する必要はない。

【0081】図9に示されたテクセル・アクセス回路の動作を図10および図11を参照しながら以下に更に例示する。図10は、上位MIPマップの複数のテクセルおよび（サイズの小さい）下位MIPマップの複数の

テクセルを示す。テクセルは、上記図8の場合に使用した標記法と同様に、 A_n 、 B_n 、 C_n および D_n というラベルをつけられている(但し n は整数)。レンダリングされるべき7つのピクセルに、 P_0 、 P_1 、 P_6 というラベルがつけられている。図に示されているように、レンダリングされるべき(複数)ピクセルは、MIPマップのテクセルに直接対応していない。この例においては、3線形補間法が実行され、上位マップから4つのテクセルおよび低位マップから4つのテクセルがアクセスされ各ピクセルごとに補間される。進行方向は、レンダリングの方向であって、ピクセルに付けられた数字番号に対応する。

【0082】図11は、キャッシュ・アクセス・コマンドFIFO(206A)、テクセル・データFIFO A0(214A0)、テクセル・データFIFO A1(214A1)およびテクセル補間回路コマンドFIFO 216を示す。他のテクセル・データ経路B、CおよびDの各々に関するFIFOも同じ形態で動作するので、便宜上テクセル・データ経路Aに関連するFIFOだけが示されている。各FIFOバッファは、データが読み書きされるべきFIFO内の単一の位置をそれぞれポイントする書き込みポインタおよび読取りポインタを含む。両ポインタは、本実施例では1回につき1位置移動することができる。

【0083】ピクセル P_0 は、上位マップにおけるテクセルA0、B0、C0およびD0に、そして下位マップにおけるテクセルA0、B0、C0およびD0に対応しているので、これら8つのテクセルが補間され、ピクセル P_0 に対するテクセル・データが生成される。ピクセル P_0 について、キャッシュから読み取られるテクセル・データをテクセル・データFIFO 214A0に書き込むべきアドレスと共に、上位マップのテクセルA0のアドレス(図10でuA0と標記されている)が、キャッシュ・アクセス・コマンドFIFO 206Aの最初の位置に書き込まれる。次に、キャッシュ・アクセス・コマンドFIFO 206Aの書き込みポインタが1位置移動され、キャッシュから読み取られるテクセル・データをテクセル・データFIFO 214A1に書き込むべきアドレスと共に、下位マップのテクセルA0のアドレス

(図10でlA0と標記されている)がFIFOの次の位置に書き込まれる。このように、上述の理由からテクセル・データFIFO 0と1は交互に使用される。キャッシュ・アクセス・コマンドFIFO 206B-206Dは低位マップのテクセルB0、C0およびD0に関して同様の方法で更新される。

【0084】ピクセル P_1 について、上位および下位マップのテクセルA1がアドレスuA1およびlA1にそれぞれ補間のため記憶される。上位および低位マップのテクセルA1は新しいテクセルであり、前のピクセル P_0 からのテクセルに対応していないので、それらはキャ

ッシュからアクセスされる。このようにして、これらテクセルのテクセル・アドレスが、それらのアドレスから読み取られるテクセル・データがテクセル・データFIFO 214A0および214A1に書き込まれるべきことをそれぞれ示す対応するアドレスと共に、キャッシュ・アクセス・コマンドFIFOの後続の2つの位置に付加される。図11は、上記情報で更新された後のキャッシュ・アクセス・コマンドFIFO 206Aを示す。

【0085】最初の2つのピクセル P_0 および P_1 については共通のAアドレス指定テクセルが存在しないので、両者に関するテクセル・データを取り出すためキャッシュ・メモリがアクセスされる。最初のコマンドがキャッシュ・アクセス・コマンドFIFO 206Aから読み取られ、アドレスuA0にあるテクセル・データがキャッシュ・メモリから読み取られテクセル・データFIFO 214A0の最初の位置に書き込まれる。そして、次のコマンドがキャッシュ・アクセス・コマンドFIFOから読み取られ、アドレスlA0にあるテクセル・データがキャッシュ・メモリから読み取られテクセル・データFIFO 214A1の最初の位置に書き込まれる。次のコマンドがキャッシュ・アクセス・コマンドFIFOから読み取られ、アドレスuA1にあるテクセル・データがキャッシュ・メモリから読み取られテクセル・データFIFO 214A0の次の位置に書き込まれる。最後に、第4番目のコマンドがキャッシュ・アクセス・コマンドFIFOから読み取られ、アドレスlA1にあるテクセル・データがキャッシュ・メモリから読み取られテクセル・データFIFO 214A1の次の位置に書き込まれる。

【0086】次のピクセル P_2 をレンダリングするため、アドレスuA1およびlA1のテクセルが補間される必要がある。これらのテクセルが前にレンダリングされたピクセル P_1 についてアクセスされたものなので、それらは、テクセル・データFIFO 214A0および214A1の最も最近書き込まれたエントリにそれぞれ記憶されている。従って、それらのテクセルについて新しいキャッシュ・アクセス・コマンドがキャッシュ・アクセス・コマンドFIFO 206Aに送られることはない。その代わり、ピクセル P_1 に関する所望のテクセル・データが補間された後、アドレスuA1およびlA1に記憶されたテクセル・データが、テクセル補間回路によってテクセル・データFIFO 214A0および214A1の最も最近読まれた位置からそれぞれアクセスされ、キャッシュへのアクセスは必要とされない。直接FIFOバッファからデータを読み取る方が、キャッシュ・メモリからデータをアクセスする場合に比較して所要時間は少ない。従って、キャッシュ・アクセスを減らす本発明のFIFOバッファはシステムの帯域幅を増加させる。

【0087】上述のように、インターリーブA-Dの各

々に対応するテクセル・データFIFO214は、別々に制御されるFIFOゼロおよび1を含む。FIFOは、このような形態で3線形補間を実行するために能率的に分割される。上述から理解されるように、上記の実施形態において、テクセル・データFIFO214の各々は、後続の読取りが同じエントリをポイントできるポインタを維持することによって、その最も最近読み取られたエントリへのアクセスを提供する。このため、連続的な読取りサイクルの間、各インターリーブが2つのマップの間で交互に読み取りを行うが、独立したFIFOが、1つのマップ内で連続的な読み取りを実行することができるので、FIFOへの連続アクセスにおいて読取りポインタが同じテクセル・データをポイントすることが可能となる。

【0088】各ピクセルがタイル作成/境界検査器72によって処理されコマンドがキャッシュ・アクセス・コマンドFIFOに送られる場合、コマンドはまたテクセル補間回路コマンドFIFO216に書き込まれる。例えば、ピクセルP0についてアドレスuA0でテクセルにアクセスすべきコマンドがキャッシュ・アクセス・コマンドFIFOに送られると、コマンドNew0が、テクセル補間回路コマンドFIFO216の最初の位置に送られる。コマンドNew0は、インターリーブAからの次のテクセル・データがキャッシュからアクセスされてテクセル・データFIFO214A0に渡されること、および、FIFOからテクセル・データを読むためにテクセル補間回路は最も最近読み取られた位置から1位置FIFO読取りポインタを移動させなければならないことをテクセル補間回路に示す。

【0089】キャッシュ・アクセス・コマンドFIFOに送られるテクセル・アドレスIA0に対応する次のコマンドについて、コマンドNew1が、テクセル補間回路コマンドFIFOの次の位置に書き込まれる。コマンドNew1は、インターリーブAからの次のテクセル・データも新しいもので、テクセル・データ補間回路214A1から読み取らなければならないことをテクセル補間回路に示す。同様に、ピクセルP1に対応するテクセル・アドレスuA1およびIA1に関連するコマンドに関して、コマンドNew0およびNew1がそれぞれテクセル補間器コマンドFIFO216の次の2つの位置に書き込まれる。

【0090】ピクセルP2については、アドレスuA1およびIA1のテクセル・データが前のピクセルP1のためFIFOに書かれたデータと同一であるので、テクセル補間器コマンドFIFO216の次の2つの位置に書かれるコマンドはO1d0およびO1d1であり、次のテクセル・データはテクセル・データFIFO214A0および214A1の最も最近読まれた位置から再読み取りされるべきであることをテクセル補間器に対してそれぞれ標示する。O1d0およびO1d1コマンド

は、FIFOから次のテクセル・データを読むため最も最近読み取られた位置からFIFO読取りポインタを移動する必要がないことをテクセル補間回路に示す。

【0091】図10は、次の3つのテーブルをリストしている。第1のテーブルはピクセルの各々について補間される必要があるテクセルを示し、第2のテーブルは、テクセル・データFIFOA0、B0、C0およびD0に記憶される必要がある別々のテクセル・データ値をリストし、第3のテーブルは、テクセル・データFIFOA1、B1、C1およびD1に記憶される必要がある別々のテクセル・データ値をリストする。ブランク空間は、キャッシュから再び読まれる必要がなく、FIFOからアクセスされることができるようキャッシュから既に読み込まれた共通テクセル・データを示す。図が示す通り、複数のピクセルについて所望のテクセル・データが補間される時、本発明のFIFO方式によって多数のキャッシュ・アクセスが節約され、この結果システムの帯域幅が増大する。

【0092】図12は、各インターリーブにおいて、あるピクセルについて読み取られるべきテクセル・データが最も最近レンダリングされたピクセルについて読み込まれたか否かを判断するため、テクスチャ・マッピング・チップによって使用される回路のブロック図である。この回路は、新しいデータをキャッシュから読み取るように指示する新しいコマンドをキャッシュ・アクセス・コマンドFIFOの1つに書き込むべきか、あるいは、テクセル・データは古いのでテクセル・データFIFOの1つから読み取られるべきことを示すコマンドをテクセル補間器コマンドFIFOに書き込むべきか判断するために使用される。図12は、インターリーブAに対応する1つの回路だけを示しているが、インターリーブB、CおよびDに対して、同様の回路が用意される。この回路は、タイル作成/境界検査器の最適化エレメントの範囲内に配置される。補間されるべき各テクセルについてタイル作成/境界検査器によって受け取られる補間されたS、T値を用いて、最適化エレメントは、バス220A上に（ブロック・タグおよびテクセル・アドレスを含む）テクセル・アドレスを出力する。テクセル・データFIFO214A0および214A1に割り当てられた最も最近処理されたテクセルのアドレスは、アドレス・レジスタ222A0および222A1にそれぞれ記憶されている。現在のテクセル・アドレスが、比較器224A0および224A1によって、レジスタ222A0および222A1に記憶されているテクセル・アドレスとそれぞれ比較される。

【0093】現在のテクセル・アドレスが、レジスタ222A0および222A1に記憶されているアドレスのいずれとも一致しない場合、そのテクセル・アドレスに対応するテクセル・データが、キャッシュ・メモリからアクセスされる必要があり、適切なコマンドがキャッシ

10

20

30

40

50

ユ・アクセス・コマンドFIFOに書かれる。しかし、テクセル・アドレスが、レジスタ222A0および222A1に記憶されているアドレスと一致する場合、テクセル・データはテクセル・データFIFO212A0または212A1にそれぞれ記憶されていて、そのアドレスに対応するテクセル・データをアクセスする直前にテクセル補間器によって読み取られる。従って、キャッシュ・アクセス・コマンドはキャッシュ・アクセス・コマンドFIFOに書かれず、テクセル・データは古いので、読取りポインタを動かすことなく最も最近読まれたFIFO位置からアクセスされるべきことを示すコマンドが、対応するテクセル補間器コマンドFIFOに書き込まれる。

【0094】V. テクスチャ・データ・ブロックの構成

図1は、8×8テクセルの基本マップ100を含む一連の平方テクスチャMIPマップを示す。基本マップを基に、サイズの的にフィルタして、最小サイズのマップ108まで連続的マップの各々が作成される。最小サイズのマップ108にはマップ番号ゼロが割り当てられ、サイズが大きくなるマップ毎に番号を1ずつ増分する。従って、本例の場合の基本マップ100はマップ番号3を持つ。マップ番号は、後述する方法で、テクスチャ・データの各ブロックに対するブロック・タグを決定する際に使用される。このマップ番号付け方式に従って、1×1テクスチャ基本マップを仮定すると、マップ番号10は1024×1024テクセルのマップに対応し、マップ番号9は512×512テクセルのマップに、マップ番号8は256×256テクセルのマップにというようにそれぞれ対応する。テクスチャ基本マップが1×1でなければ、マップ番号10は、1024テクセルより大きい次元を持つマップに対応する。ここでの記述は、正方形のテクスチャ・ベース・マップを仮定しているが、長方形のマップも可能である。長方形の場合、マップ番号は、マップの長い方の次元のテクセル数によって決定される。例えば、マップ番号10を持つ長方形のマップは、長次元に1024以上のテクセルを持つ。上記以外のマップ番号付け法を使用できる点は理解されるべきであろう。

【0095】マップ番号10を持つ正方形1024×1024テクセル・マップは、マップ内の各テクセル位置をユニークに識別するため10ビットのS座標S[9:0]および10ビットのT座標T[9:0]を必要とする。同様に、マップ番号9を持つマップは、マップ内の各テクセル位置を識別するため9ビットのSおよびT座標を必要とし、マップ番号8を持つマップは、マップ内の各テクセル位置を識別するため8ビットのSおよびT座標を必要とするというように、以下のマップ番号について同様となる。任意のピクセルに対応するMIPマップのテクセルの位置をユニークに識別するSおよびT座標は上述の方法で補間される。

【0096】詳細は後述するが、テクスチャ・データは、(図2の)ホスト・コンピュータ15の主メモリ17に256×256テクセルのブロックの形式で記憶される。キャッシュ・ミスが発生すると、キャッシュ・ミスのあったテクスチャ・データのブロックを識別するキャッシュ・タグが、ホスト・コンピュータによって読み取られ、次に、そのブロックのテクスチャ・データがテクスチャ・マッピング基板のキャッシュ・メモリ48へダウンロードされる。本発明の実施形態において、任意の1時点で、64ブロックのテクスチャ・データがキャッシュ・メモリに記憶されることができる。これらの64ブロックのテクスチャ・データは、1つまたは複数のテクスチャの複数のMIPマップからのデータを含むことができる。各ブロックは、それをユニークに識別するブロック・タグを持つ。9以上のマップ番号を持つMIPマップは、256×256を超えるテクセルを含み、従って、複数のブロックの形態で記憶される。複数ブロックの形態で記憶されるマップに対する高位S、T座標は、マップを記憶するデータ・ブロックに関するブロック・タグに含められる。

【0097】例えば、マップ番号9を持つMIPマップは、512のテクセルに等しい1つの次元を持ち、正方形の場合は、サイズの的に512×512テクセルである。(正方形マップを仮定すると)マップは1ブロック256×256テクセルの4つのブロックに分割される。従って、それらのブロックの各々に対するブロック・タグは、マップの範囲内でのブロックの位置を識別する1つの高位S座標ビット(S[8])および1つの高位T座標ビット(T[8])を含む。同様に、マップ番号10を持つMIPマップはサイズの的に1024×1024テクセルであり、1ブロック256×256テクセルの16のブロックに分割される。従って、それらのブロックの各々に対するブロック・タグは、マップの範囲内でのブロックの位置を識別する2つの高位S座標ビット(S[9:8])および2つの高位T座標ビット(T[9:8])を含む。

【0098】後述するが、補間の間システムの帯域幅を減らすため、隣接するMIPマップの同じ部分が反対側のSDRAMバンクに記憶されるように、テクスチャMIPマップは更に小さく分割されてメモリに記憶される。加えて、キャッシュ・メモリ内のメモリ空間を効率的に利用するため、256×256未満テクセルの複数マップをキャッシュ・メモリの1つのブロックの中に記憶することができる。

【0099】図13は、

LA

95

という面画像を含む特定テクスチャに関する一組のテクスチャMIPマップを示す。図13に示されるように、あるテクスチャに関する一連のMIPマップにおけるM

I Pマップの各々は、1つの平方テクスチャ・マップに対して等しいサイズの4つの象限に分割される。図12に示される例においては、基本マップは、マップ番号9を持ち、(画像Lを含む)9Q1、(画像Aを含む)9Q2、(画像9を含む)9Q3および(画像5を含む)9Q4の象限に分割されている。同様にマップ番号8は、それぞれL、A、9および5を含む象限8Q1、8Q2、8Q3および8Q4に分割されている。同様にマップ番号7は、それぞれL、A、9および5を含む象限7Q1、7Q2、7Q3および7Q4に分割されている。同様に、更に小さいマップは同様の象限に小分割されている。

【0100】各MIPマップの2つの象限が、キャッシュを形成するSDRAMの1つのバンクに記憶され、残りの2つの象限が反対側のバンクに記憶される。本発明のテクスチャ・データ配置方式に従えば、8以上の番号の(すなわちサイズが256×256テクセル以上の)基本マップを持つテクスチャに関しては、そのテクスチャのMIPマップすべての象限のすべてについてメモリ空間のブロック内のメモリ位置はあらかじめ定められている。例えば、図14に示されるように、マップ番号9の象限9Q1および9Q4は、キャッシュ・バンク1内の別々のブロックに記憶され、象限9Q2および9Q3は、キャッシュ・バンク0内の別々のブロックに記憶される。隣接するMIPマップの対応する象限は、反対側のバンク内のブロックに記憶される。この例において、それぞれ象限9Q1および9Q4をフィルタしたデータを含む象限8Q1および8Q4は、キャッシュ・バンク0内の同じブロックに記憶される。同様に、それぞれ象限9Q2および9Q3をフィルタしたデータを含む象限8Q2および8Q3は、キャッシュ・バンク1内の同じブロックに記憶される。図14は、図13に対してスケールが合うように描かれてはいない。図13のマップの象限が、対応する図14のものと同じ大きさであることは理解されなければならない。

【0101】マップのそれぞれのサイズに従って、マップ番号9の各象限は完全な256×256テクセル・ブロックを占めるが、マップ番号8の4象限は各々ブロックの1/4だけを占める。従って、象限8Q2および8Q3は合わせて同じブロックの1/2を占め、象限8Q1および8Q4は、反対のバンク内のもう1つのブロックの1/2を占める。キャッシュ・メモリ空間を効率的に割り当てるため、それらブロックの各々の中で空いている位置は、マップ番号7以下の適切な象限によって占められる。従って、ゼロないし8の番号を持つマップのすべては、それぞれ別のバンクにある2つのブロックを占める。

【0102】8以下のマップ番号を持つマップに関する4象限の位置は、(8以上のマップ番号を持つ基本マップを所与として)、図14に示される形態にあらかじめ

定められる。図に示されているように、右上の象限8Q2および左下象限8Q3は同じ物理的関係を維持して、それぞれ第1のブロックの右上および左下の象限を占め、左上の象限8Q1および右下象限8Q4も同じ物理的関係を維持して、第1のブロックとは異なるバンクにある第2のブロックの左上および右下の象限をそれぞれ占めている。また、象限7Q1および象限7Q4は同じ物理的関係を維持して、それぞれ第1のブロックの左上の象限を占め、象限7Q2および象限7Q3は同じ物理的関係を維持して、第2のブロックの右上の象限をそれぞれ占めている。

【0103】3線形補間の間、1つのピクセルが、1つのMIPマップの中の4つのテクセルと隣接するMIPマップの中の4つのテクセルの間にあるテクスチャ・マップの位置に対応すれば、すべての8つのテクセルがキャッシュからアクセスされる。両方のMIPマップからアクセスされるテクセルは、大きい方のマップのデータをフィルタリングした小さい方のマップのデータと共に、共通のテクスチャ・データを含む。上述のように、オブジェクトのピクセルがレンダリングされる時、隣接ピクセルは、そのテクスチャについて同じ2つのMIPマップに対応することがしばしばあり、2つのマップを記憶するキャッシュ・ブロックの間でキャッシュへの読取りを連続的に切り換える必要が生じる。キャッシュSDRAMチップの異なるバンクに隣接MIPの共通データを記憶することによって、連続的な読取りサイクルの間2つのMIPマップの間でのキャッシュ読取り切り換えによる再ページングの負荷が発生しない。これは、3線形補間の効率的な実施を提供する。

【0104】上述の説明から理解されるように、テクスチャが8以上のマップ番号を持つ基本マップを含む場合、そのテクスチャに対するブロック間のMIPマップ割り付けは、本発明の上述の実施例に従って、あらかじめ定められている。すなわち、マップ番号8を持つマップの2つの象限が、図14に関して上述したように、バンクの1つの範囲内の第1のブロックの予め定められた位置を占め、マップ番号8を持つマップの別の2つの象限が、反対バンクの別の1つのブロックの範囲内の予め定められた反対の位置を占める。しかし、マップ番号7以下の基本マップを持つテクスチャについては、(各バンクに1つのブロックの)2つのブロック内の複数の位置がマップを記憶するために使用可能であり、ホスト・コンピュータによって選択される。複数のマップ部分が単一ブロックのデータを共有する時、共有されたブロック内の各マップの位置を識別するため、以下に記述される方法で、サブテクスチャ識別子(ID)が割り当てられる。

【0105】図13の一連のMIPマップの構成に加えて、図14は、異なるテクスチャからの第2の一連のMIPマップ(図でチェッカー盤模様部分)がメモリ・ブ

ロックの間に割り当てられる。この第2のテクスチャのMIPマップは小分割され、第1のテクスチャと同じ方法で別々のブロックに記憶される。図14の構成が別々のブロックに構成される異なるテクスチャのMIPマップを示してはいるが、2つの異なるテクスチャからのテクスチャ・データを同じブロック内に記憶することもできる点は理解されるべきであろう。

【0106】上述のとおり、本発明の1つ実施形態において、キャッシュ・メモリは、テクスチャ・マッピング・データの最高64までのブロック（各ブロックは256×256テクセルを含む）を記憶することができる。キャッシュ・メモリは、ブロック0-31を収納するバンク0およびブロック32-63を収納するバンク1という2つのバンクに区分される。キャッシュ・ディレクトリは、キャッシュのブロックに対応する最高64までのブロック・タグ・エントリを含む。キャッシュ・ディレクトリ内の各ブロック・タグの物理的な位置は、キャッシュ・メモリ内のテクスチャ・データの対応するブロックの物理的な位置を識別する。ブロックの位置を示すブロック・タグから、ブロック・インデックスが生成される。キャッシュのテクセルに関するキャッシュ・アドレスは、ブロックに対するブロック・インデックスおよびキャッシュ・メモリ内のテクセル・アドレスによって形成される。テクセル・アドレスは、テクセルに関する補間された低位S、T座標を含み、また場合によっては以下に述べるようにサブテクスチャIDのビットを含む。

【0107】図15は、4象限に小区分されているマップ番号9を持つテクスチャMIPマップの1例を示す。MIPマップは512×512テクセルであり、従って、各象限はサイズ256×256テクセルでありメモリの1ブロックに対応する。本発明の1つの実施形態に従って、MIPマップの各象限に割り当てられるべきキャッシュ・バンクを決定する簡単な方式がホスト・コンピュータによって実施される。MIPマップの4象限の各々について、象限に関するSおよびT座標の最上位ビットの値に対する論理的排他OR演算の結果が、象限が割り当てられるキャッシュSDRAMバンクを指し示す。

【0108】512×512テクセルのマップについては、9つのS座標ビットS[8:0]および9つのT座標ビットT[8:0]がマップ内の各テクセルの位置を指定する。象限境界は、SおよびT座標ビットS[8]およびT[8]によって表されるSおよびT次元両方の中間点に定められる。従って、マップ番号9を持つMIPマップの4つの象限の各々に関するキャッシュ・バンクを決定するため、各象限の対応する最上位SおよびT座標ビットS[8]およびT[8]の値に対する論理的排他OR演算が実行される。同様に、マップ番号10を持つMIPマップに関しては、その4つの象限の各々

に関するキャッシュ・バンクは、各象限の対応する最上位SおよびT座標ビットS[9]およびT[9]の値に対する論理的排他OR演算によって決定される。奇数のマップ番号を持つMIPマップについては、隣接マップからの共通データが異なるバンクに記憶されるようにするため排他OR演算の結果が反転される。

【0109】図15で示される例において、ブロック1ないしブロック4は、それぞれ、左上象限、右上象限、左下象限および右下象限の512×512テクセル・マップに対応する。ブロック1ないしブロック4について、ビットS[8]、T[8]はそれぞれ[0,0]、[1,0]、[0,1]および[1,1]に等しい。従って、ブロック1についてXORS[8]XORT[8]演算の結果はゼロとなる。マップが奇数マップ番号（すなわち9）を持つので、この結果の反転値（すなわち1）によって、ブロック1はキャッシュ・バンク1に記憶されるべきことが標示される。ブロック2については、S[8]XORT[8]演算の結果の反転がゼロであって、ブロック2はキャッシュ・バンク0に記憶されるべきことが標示される。ブロック3およびブロック4については、S[8]XORT[8]演算の結果の反転がそれぞれ1およびゼロであって、ブロック3はキャッシュ・バンク1に、ブロック4はキャッシュ・バンク0にそれぞれ記憶されるべきことが標示される。

【0110】図15の例で示されているものと同じテクスチャについてマップ番号10を持つマップに関する限り、そのマップのサイズが1024×1024テクセルであるため16個の256×256テクセル・ブロックに区分けされる。各ブロック毎に、S[9]XORT[9]演算の結果がその特定ブロックに対するバンク番号を標示する。マップ番号10を持つマップの各ブロック毎のXOR演算の結果は、マップ番号9を持つ隣接マップの場合のように反転されずに、これら2つの対応する象限は異なるキャッシュ・バンクに記憶される。

【0111】マップのサイズに応じて、マップを表すテクスチャ・データ・ブロックのブロック・タグは、特定のMIPマップ内のブロックの位置を示す少なくとも1つの高位S座標ビットおよび高位T座標ビットを含む。マップ番号9を持つ512×512テクセルMIPマップについては、MIPマップ内の各ブロックの位置を示すためにブロック・タグ内にただ1つのS座標ビットおよびT座標ビットが必要とされる。マップ番号10を持ち、16ブロックのデータを含む1024×1024テクセルMIPマップについては、MIPマップ内の各ブロックの位置を示すためにブロック・タグ内に2つのS座標ビットおよびT座標ビットが必要とされる。8以下のマップ番号を持つマップに関する限り、ブロック・タグにSおよびTビットは必要とされない。テクスチャMIPマップ・データをホスト・コンピュータの主メモリからキャッシュ・メモリへダウンロードする際、ホスト

・コンピュータは、上述の排他的OR演算方式を使用してブロック・タグの上位SおよびT座標ビットをデコードして、各データ・ブロックが書き込まれるべき特定バンクを決定する。

【0112】未使用メモリ空間を最小にするようにテクスチャ・データを割り当てるために、各データ・ブロックは、1サブブロックが64×64テクセルである16個のサブブロックにさらに小区分される。テクスチャ・データの各サブブロックは、ブロック内の特定サブブロックの位置を識別するサブテクスチャIDを含む。サブテクスチャIDは、2つのSビットS[1:0]および2つのTビットT[1:0]を含む。1つまたは複数のテクスチャの1つまたは複数MIPマップからの複数サブテクスチャを1つのブロックに記憶することも可能である。

【0113】図16において、ブロック1およびブロック2が、各々16個の64×64テクセル・サブブロックに小区分されているキャッシュのバンク0および1にそれぞれ割り当てられている。各ブロックのサブテクスチャは、ST0ないしST15という符号をつけられ、2つのS座標ビットおよび2つのT座標ビットを含むサブテクスチャIDによって識別される。サブテクスチャは、上述のメモリ割り当て方式と整合性がとれるように、一貫した符号が付けられるが2つのキャッシュ・バンク内で鏡面反射位置を持つ。64×64テクセルのサブテクスチャのサイズは例示のため選択したもので、変えることはできる。一層小さいサイズのサブテクスチャは同じブロック内に更に多くのテクスチャを詰め込むことができる。サブテクスチャのサイズを小さくすればサブテクスチャIDが一層多くのビット必要とする点は理解されるべきであろう。

【0114】レンダリングの間、一連のテクセルを補間するため、テクスチャID、サブテクスチャIDおよび当該テクスチャに関する基本マップのサイズを表す8ビット・ワードが、3Dパイプラインを経由して、それらデータを20ビット・レジスタ（図示されてない）に一時的に記憶するタイル作成／境界検査器へ送られる。補間されるべきテクセルが異なるサブテクスチャIDまたはテクスチャIDを持つ場合、新しいデータがタイル作成／境界検査器へ送られ、レジスタに記憶される。サブテクスチャIDは以下に述べるようにテクセル・アドレスの一部として使用される。

【0115】テクセル・アドレスがサブテクスチャIDの下位S、T座標ビットを含むか否かは、アドレス指定されているマップのサイズおよびそのテクスチャの基本マップのサイズに依存する。アドレス指定されているマップが、7以下のマップ・サイズであり、また、その対応する基本マップもまた7以下のサイズである場合、以下に詳細を説明するように、ブロック内のサブテクスチャの位置のアドレスを示すため、テクセル・アドレスの

特定の上位アドレス・ビットが、サブテクスチャIDのビットを含む。上述のように、基本マップがマップ番号8以上をもつ場合、それぞれのデータ・ブロックの範囲内のそのテクスチャに関するMIPマップ象限のすべての位置はあらかじめ定義されている。従って、マップ番号8以上を持つマップの1つからテクスチャが取り出される時、サブテクスチャを使用せず、既知のあらかじめ定められた位置を使用して各象限に関するテクセル・アドレスの上位ビットが生成される。しかし、テクスチャの基本マップが7以下のマップ番号を持つ時、MIPマップ象限の位置はあらかじめ定められてなく、サブテクスチャIDビットをテクセル・アドレスの上位ビットとして使用してサブテクスチャの位置を決定する。

【0116】上述のように、異なるテクスチャからの複数のマップは、テクスチャの基本マップが十分小さい限り、単一のデータ・ブロックの異なるサブテクスチャ内に記憶することができる。この場合、各マップについてのテクスチャ・アドレスが下位テクスチャIDビットを含む。例えば、4つの異なるテクスチャからのマップ番号7を持つ4つの異なるマップは、1つのブロック内で異なるサブテクスチャ内に割り当てられ、各テクスチャの基本マップのマップ番号が7である場合、サブテクスチャIDの1S座標ビットおよび1T座標ビットが、テクスチャを見分けるテクセル・アドレスの一部である。タイル作成／境界検査器がテクセル・アドレスを計算するルーチンは、図18を参照して後述される。

【0117】本発明の実施形態において、テクスチャMIPマップ・データは、一度に1ブロックずつダウンロードされる。しかし、サブテクスチャが主メモリからダウンロードされることができるようサブテクスチャIDをブロック・タグに含めることができる点は理解されるべきであろう。また、本実施形態で記述されるブロックおよびサブテクスチャのサイズは単に例示の目的のものにすぎず、アプリケーションにとって都合のよいように変更することは可能である。

【0118】VI. キャッシュ・ブロック・タグおよびブロック・インデックス

キャッシュ・ディレクトリは、64個のエントリの各々に関するブロック・タグを含み、各エントリ毎に対応するブロック・インデックスを識別する。ブロック・インデックスは、テクスチャ・データの対応するブロックの先頭が記憶されるキャッシュの物理的な位置を識別する。ブロック・タグは、図17に示される方法でテクスチャ・データの各ブロックをユニークに識別する23ビットの識別子である。

【0119】テクスチャ・データの任意のテクセルをユニークに識別するため、そのテクセルが対応するテクスチャが識別されなければならない。本発明の1つの実施形態において、テクスチャ・マッピング・ハードウェアは、1つのテクスチャをユニークに識別する8ビット・

テキストチャIDを導入する。加えて、同じブロック内に記憶される異なるテキストチャからのテキストチャ・データについて、4ビットのサブテキストチャIDが、テキストチャを識別するハードウェアによってサポートされる。このように、本発明のテキストチャ・マッピング・ハードウェアは、任意の1時点で使用可能な2¹⁶すなわち4096個のユニークなテキストチャをサポートする。

【0120】上述のとおり、各テキストチャは、一連のMIPマップによって表現され、本発明の1つの実施形態において、MIPマップの各々は、一連のMIPマップにおける位置を示すマップ番号を備えている。このように、任意のテクセル・データは、そのテキストチャに関するテキストチャID、サブテキストチャIDおよび基本マップのサイズによって識別されるだけでなく、それが対応するMIPマップのマップ番号によっても識別される。最後に、テクセルは、そのSおよびT座標（すなわちその補間されたS、T値）によって、MIPマップ内でユニークに識別される。

【0121】サブテキストチャIDおよびテキストチャ・マップ基本サイズの他、テクセルをユニークに識別する上記のパラメータ類を使用して23ビットのブロック・タグが生成される。マップ番号およびSならびにT座標に関しては、本発明の1つの実施形態において、SならびにT座標を生成するために使用されるハードウェアが15ビットに限定されている。従って、この実施形態に関する限り、ハードウェアによってサポートされる最大のテキストチャ・マップは、15ビットSフィールド[14:0]および15ビットTフィールド[14:0]を持ち、その結果、最大テキストチャ・マップは32K×32Kテクセルとなる。上述のとおり、テクセル・データの各ブロックは、256×256テクセルを含む。従って、低位SならびにTビット（すなわちT[7:0]ならびにS[7:0]）がテクセル・データ・ブロック内の特定のテクセルを識別するために使用される。高位SならびにTビット（すなわちT[14:8]ならびにS[14:8]）だけがテクセル・データの特定ブロックを識別するためブロック・タグの中で使用される。

【0122】上述のとおり、各MIPマップは、その対応するテキストチャに関する一連のマップ内でそのマップを識別するマップ番号を割り当てられる。あるテキストチャに関する一連のマップにおけるMIPマップの数にかかわらず、その中の最小の（すなわち1×1テクセルのサイズの）MIPマップにマップ番号0が割り当てられる。32K×32Kテキストチャに関する一連のMIPマップの最大のものは16個のMIPマップを含むので、サポートされる最大のMIPマップ番号は15である。

【0123】図17は、ブロック・タグが形成される様態を示す。ブロック・タグの上位8ビット[22:15]は、テキストチャ・データのブロックによって表され

るテキストチャのテキストチャIDに対応する。ブロック・タグの低位ビット[13:00]は、高位TならびにS座標[14:08]ならびにS[14:08]に対応する。ブロック・タグ[14]は、高位T座標フィールドの値と連係してマップ番号の識別を可能にするマップ・ビットに対応する。最大32K×32Kより小さいマップは、小さくなる程少ないビット数となっており、SならびにTアドレス・フィールド全体を使用しない点は理解されるべきであろう。図17に示されるように、9以上のマップ番号を持つマップについては、未使用ビット中の最下位T座標ビットに対応するブロック・タグ・ビットは、論理「0」にセットされ、残りの上位T座標ビットに対応するブロック・タグ・ビットは、論理「1」にセットされる。例えば、T座標ビットすべてを使用するマップ番号15については、マップ・ビットが論理「0」にセットされている。マップ・ビットに対応するブロック・タグ・ビット[14:07]および高位T座標ビット

[14:8]を読み取ることによって、左から右に読んで最初の論理「0」に出会う位置が、ブロック・タグによって表されるマップ番号を示す。論理「1」がブロック・タグ・ビット[14:08]のすべてに含まれている場合、そのマップ番号が8以下であることを表す。

【0124】上述のように、8以下のマップ番号を持つ特定テキストチャのマップのすべては、それぞれが別のバンクに位置する2つのデータ・ブロック内に記憶される。8以下のマップ番号を持つマップの各々の2つの象限すなわち半分が、2つのブロックの各々の中に記憶される。ブロック・タグ・ビット[07]は、8以下のマップ番号を持つマップの1/2部分の各々が2つのブロックのいずれに記憶されているかを示す。このように、8以下のマップ番号を持つマップの各々について、ブロック・タグ・ビット[07]は、そのマップの1/2がバンク・ゼロに記憶されている場合は「0」の値を持ち、バンク1に記憶されている別の1/2について値「1」を持つ。特定テキストチャからの8以下のマップ番号を持つマップのすべてが2つのブロック内に記憶されるので、それらのブロックを識別するため、1ブロック・タグ・ビットだけが使用される点は理解されるべきであろう。従って、8以下の番号を持つマップの各々に関する特定のマップ番号は、ブロック・タグ・フィールドの一部として記憶されない。

【0125】8以下の番号を持つマップの各々の各象限に関するブロック・タグ・ビット[07]の値は、当該象限が記憶されるべきバンクを決定する方式に基づいて計算される。この方式は、マップ番号が偶数の場合各象限について実行されるMSB（上位）ビット値の論理的排他OR演算であり、奇数の場合は上記演算結果の反転値である。

【0126】図17に示されるように、上位Sアドレス・ビットに対応するブロック・タグ・ビット[6:0]

は、Sアドレス・ビットが使われないマップ番号8以下の小さいマップについて論理「0」にセットされるため、論理「0」に等しくなければならないことを示すマップ番号に関連して、これらのビットのいずれかが論理「1」として検出されればキャッシュ・ディレクトリ・エントリには有効なデータが含まれていないことを示すように、上位Sアドレス・ビットを使用することができる。

【0127】各MIPマップ象限について、該象限に関する最上位SならびにT座標の値に対する論理的排他OR（すなわちXOR）演算の結果が、該象限が割り当てられるべきキャッシュのSDRAMバンクを指し示す。バンク番号は、偶数のマップ番号を持つマップについてはこのXOR演算結果に等しく、奇数のマップ番号を持つマップについてはこのXOR演算結果の反転値に等しい。これは、図17のテーブルの右欄に、XOR演算を示す記号“^”および論理反転を示す記号“!”を用いて示されている。9以上のマップ番号を持つマップに関しては、各象限は、少なくとも1つのデータ・ブロックの全体を使用し、各ブロックは、（図17の最後の欄に示される）XOR演算によって指し示されるバンクに記憶される。

【0128】8以下のマップ番号を持つマップについて、それらのマップのすべては、（各バンクに1つのブロックという形態で）2つのデータ・ブロックを占める。図17のテーブルの最後の2つの行は、8以下のマップ番号を持つマップの別々の半分（2つの象限）に対応する。ブロック・タグ・ビット[07]が、マップの半分がバンク0ブロックあるいはバンク1ブロックのいずれに記憶されるかを表す。ビット[07]の値は、上述のXOR演算に基づいて計算される。例えばマップ番号8を持つマップの場合、マップの象限の各々について、ブロック・タグ・ビット[07]は、S[7] XOR T[7]に等しい。マップ番号7を持つマップの象限の各々について、ブロック・タグ・ビット[07]は、S[6] XOR T[6]の反転値に等しい。7より小さいマップ番号を持つマップの各象限に関するブロック・タグ・ビット[07]は、同様に、番号の奇偶に応じて計算される。（8以下のマップ番号を持つ）マップ各々の2つの象限は同じブロックに記憶されるため、各マップのそれらの2つの象限が同じブロック・タグ・ビット[07]を持つこととなる点は理解されるべきであろう。

【0129】（アクセスされるべきテクセルをアドレスする）補間されたS、T座標とキャッシュ・ディレクトリの23ビット・ブロック・タグの中の1つの間でヒット（一致）が発生すると、キャッシュ・ディレクトリは、そのテクセルを含むキャッシュ・ブロックが記憶されているキャッシュ・メモリの物理的な位置を識別するブロック・インデックスを生成する。キャッシュは、任

意の1時点で64ブロックのテクセル・データを記憶する。従って、キャッシュ・メモリにおけるブロック・アドレスを識別するため、先に述べたように、キャッシュに対する高位アドレス・ビットの役目を果たす6ビットのブロック・インデックス（ $2^6=64$ ）が提供される。

【0130】テクセル・アドレスは、 256×256 テクセル・ブロック内でアクセスされるべきテクセルの位置を示すビットS[7:0]およびT[7:0]を含む16ビット・ワードである。テクセル・アドレスは、補間されたS、T座標、アクセスされるべきマップのマップ番号、テクスチャならびにサブテクスチャID、およびテクスチャの基本マップのサイズを使用して、図18を参照して記述されるルーチンに従って計算される。上述のように、テクセルが記憶される該当するインターリーブを決定するため、テクセル・アドレスの下位（LSB）Sビットおよび下位（LSB）Tビットがデコードされる。テクセル・アドレスの残りの14ビットは、（キャッシュ・アドレスの6つのMSBすなわち上位ビットである）6つのブロック・インデックス・ビットと連係して、デコードされたキャッシュ・インターリーブ内のSDRAMペアに送られるキャッシュ・アドレスとして使用される。

【0131】VII. テクセル・アドレス計算

レンダリングの間、タイル作成/境界検査器エレメント72は、パラメータ補間器64から、アクセスされるべきテクセルの補間されたS、T値およびテクセルがアクセスされるべきマップのマップ番号を表す4ビット・ワードを受け取る。パラメータ補間器64から受け取る補間されたS、T座標値の各々は、16個の整数ビットおよび8個の小数ビットを含む。マップ番号を表す4ビット・ワードは、（テクセル・サイズ1の）マップ番号0から（ $32k \times 32k$ テクセル・サイズの）マップ番号15に至る範囲を含み、既に記述したように勾配から計算される。次に、補間されたS、T値とキャッシュ・ディレクトリにおけるブロック・タグ・エントリの比較が実行される。ブロック・タグの1つとのヒットが発生すれば、ブロック・インデックスが生成される。キャッシュ・ディレクトリ・サーチが実行されている時間と並列して、図18を参照して記述されるルーチンに従ってテクセル・アドレスが計算される。

【0132】テクセル・アドレスは、タイル作成/境界検査器によって、テクセルのテクスチャID、サブテクスチャID、マップ番号、基本マップ番号および補間されたS、T座標を用いて計算される。タイル作成/境界検査器はこれらの情報のすべてを持つ。アクセスされるべきユニークなテクセル毎に、タイル作成/境界検査器は、パラメータ補間器から、（S、Tの各々について16個の整数ビットと8個の小数ビットを含む）補間されたS、T座標およびテクセルがアクセスされるべきマッ

ブ番号を表す4ビット・ワードを受け取り、更に、(パラメータ補間器を通過して来る) 3Dパイプラインを経由して、8ビット・テクスチャID、4ビット・サブテクスチャIDおよび該テクスチャに対する基本マップのサイズを表す8ビット・ワードを含むコマンドを受け取る。基本マップのサイズを表す8ビット・ワードは、本発明のマップ番号づけ方式に対応し基本マップのS軸とT軸のサイズをそれぞれ定義する4つのSビットならび4つのTビットを含む。例えば、4ビットのSおよびTワードの各々は、(1テクセル次元に対応する)ゼロから(32kのテクセルの次元に対応する)15に至る範囲の値を持つことができる。テクスチャID、サブテクスチャIDおよび基本マップ番号を含む20ビット・データが、キャッシュからアクセスされるべき次のテクセルに関する新しく異なるデータと置き換えられるまで、タイル作成/境界検査器内に配置される(図示されていない)20ビット・レジスタに一時的に記憶される。この情報を使用して、タイル作成/境界検査器は、各テクセル毎のテクセル・アドレスを計算する。

【0133】上述のように、(256×256テクセルの基本マップ以上に対応する)8以上のマップ番号の基本マップを持つテクスチャについては、そのテクスチャ内の各マップの象限は、テクスチャ・データ・ブロックおよびキャッシュ・メモリ・バンク内のあらかじめ定められた位置を持つ。そのようなテクスチャのテクセルに関するテクセル・アドレスの各ビットは、既知のあらかじめ定められた割り当て方式に従って計算される。しかし、(128×128テクセルの基本マップ以下に対応する)7以下のマップ番号の基本マップを持つテクスチャについては、そのテクスチャの複数マップの各象限について多数のメモリ位置が使用可能であるので、テクセル・アドレスの一定の上位ビットが、サブテクスチャIDのビットのすべてまたは一部を含む必要がある。

【0134】テクセル・アドレスを計算するためにタイル作成/境界検査器によって実施されるルーチンが図18の流れ図によって示される。ルーチンは、完了するため1サイクルを必要とする。ルーチンは、テクスチャ・マッピング・チップの境界検査器部分を形成する一組の論理ゲート(図示されていない)によって実施されることができる。図18によって概要が示されているルーチンを実行する論理ゲートを実施する方法は当業者によって理解されるべきものであろう。例えば、このルーチンをVerilogのようなソフトウェア・シミュレーション言語で書き、メイン・プロセッサ上で動くSynopsysのような合成ツールによって論理ゲート回路に変換することが可能である。その代替方法として、このルーチンをソフトウェアで書きプロセッサによって実行することもできる

ルーチンはステップ250で開始し、テクセル・アドレス・ビットS[7:0]およびT[7:0]が、補間さ

れたS、T座標ビットS[7:0]およびT[7:0]に等しくなるように事前設定される。このステップで事前設定された値は、後にリセットされない限りそのままの値が維持される。次に、補間されたテクセルが記憶されている特定マップが8以上のマップ番号を持つかが判断される(ステップ262)。もしそうであれば、そのようなテクセルに関する限りこのルーチンは終了し、テクセル・アドレスに関するビット値は、事前設定された補間されたS、T座標のまま残る。

10 【0135】マップ番号が8未満であれば、ルーチンはステップ254に進み、テクセルがバンク番号1またはバンク番号0のいずれに記憶されているかが決定される。上述のように、ブロック・タグ・ビット[07]の値を検査することで、どちらのバンクにテクセルが記憶されているかはわかる。

【0136】テクセルがバンク番号1に記憶されている場合、ルーチンはステップ256へ進み、一定のテクセル・アドレス・ビットをその事前設定値からリセットする。マップ番号1ないし4を持つマップについては、テクセル・アドレス・ビットS[4]=1、マップ番号1および2を持つマップについては、テクセル・アドレス・ビットS[2]=1とする。テクセルがバンク0に記憶されている場合、ルーチンはステップ258へ進み、マップ番号0ないし5を持つマップについては、テクセル・アドレス・ビットS[5]=1とし、マップ番号0ないし3を持つマップについては、テクセル・アドレス・ビットS[3]=1とし、マップ番号0および1を持つマップについては、テクセル・アドレス・ビットS[1]=1とする。

30 【0137】ステップ256、ステップ258いずれの後もステップ260へ進み、基本マップが8以上のマップ番号を持つかが判断される。そうであれば、ステップ262へ進み、テクセルがバンク0または1のいずれに記憶されているかが判断される。テクセルがバンク1に記憶されている場合、ルーチンはステップ264へ進み、マップ番号7を持つマップについては、テクセル・アドレス・ビットS[7]=0とし、マップ番号0ないし6を持つマップについては、テクセル・アドレス・ビットS[7:6]=0:1とする。次に、ルーチンはそのようなテクセルについて終了する。バンク0に記憶されているテクセルの場合、ルーチンはステップ266へ進み、マップ番号7を持つマップについては、テクセル・アドレス・ビットS[7]=1とし、マップ番号0ないし6を持つマップについては、テクセル・アドレス・ビットS[7:6]=1:0とする。次に、ルーチンはそのようなテクセルについて終了する。

40 【0138】基本マップが8以上のマップ番号を持っていない場合、ルーチンはステップ268へ進み、基本マップが7に等しいマップ番号を持っているか否かが判断される。そうであれば、ステップ270へ進み、テクセル

がバンク0または1のいずれに記憶されているかが判断される。テクセルがバンク1に記憶されている場合、ルーチンはステップ272へ進み、マップ番号7を持つマップについては、テクセル・アドレス・ビットS[7]をサブテクスチャIDビットS[1]の反転に等しく、テクセル・アドレス・ビットT[7]をサブテクスチャIDビットT[1]に等しくし、マップ番号0ないし6を持つマップについては、テクセル・アドレス・ビットS[7:6]をサブテクスチャIDビットS[1]の反転および1にそれぞれ等しくさせ、テクセル・アドレス・ビットT[7]をサブテクスチャIDビットT[1]に等しくする。次に、ルーチンはそのようなテクセルについて終了する。テクセルがバンク0に記憶されている場合、ルーチンはステップ274へ進み、マップ番号7を持つマップについては、テクセル・アドレス・ビットS[7]をサブテクスチャIDビットS[1]に等しく、テクセル・アドレス・ビットT[7]をサブテクスチャIDビットT[1]に等しくし、マップ番号0ないし6を持つマップについては、テクセル・アドレス・ビットS[7:6]をサブテクスチャIDビットS[1]および0にそれぞれ等しくさせ、テクセル・アドレス・ビットT[7]をサブテクスチャIDビットT[1]に等しくする。次に、ルーチンはそのようなテクセルについて終了する。

【0139】(ステップ260において) テクスチャの基本マップが8以上のマップ番号を持ってなく、かつ、(ステップ268において) マップ番号が7に等しくない場合、当然テクスチャの基本マップが6以下のマップ番号を持っているので、ルーチンはステップ276へ進み、テクセルがバンク0または1のいずれに記憶されているかが判断される。テクセルがバンク1に記憶されている場合、ルーチンはステップ278へ進み、テクセル・アドレス・ビットS[7:6]をサブテクスチャIDビットS[1:0]の反転に等しくセットし、テクセル・アドレス・ビットT[7:6]をサブテクスチャIDビットT[1:0]に等しくセットする。次に、ルーチンはそのようなテクセルについて終了する。テクセルがバンク0に記憶されている場合、ルーチンはステップ280へ進み、テクセル・アドレス・ビットS[7:6]をサブテクスチャIDビットS[1:0]に等しくセットし、テクセル・アドレス・ビットT[7:6]をサブテクスチャIDビットT[1:0]に等しくセットする。次に、ルーチンはそのようなテクセルについて終了する。

【0140】VIII. テクスチャ・データ構成の例

以下の例は、本発明の上述の実施形態に従って、ホスト・コンピュータがテクスチャ・データを構成するプロセスを説明するものである。特定のアプリケーションに関して、レンダリングされるプリミティブAがテクスチャAに対応し、プリミティブBがテクスチャBに対

応する場合がある。1つの可能性として、ホスト・コンピュータがテクスチャAを複数のテクスチャ・データ・ブロックに構成し、テクスチャBをテクスチャAと同じブロック内の異なるサブテクスチャに構成することがある。ホスト・コンピュータは、プリミティブAをレンダリングする前にテクスチャAおよびBを含むテクスチャ・データ・ブロックをキャッシュ・メモリへダウンロードする。

【0141】代替方法として、ホストはテクスチャAを複数のテクスチャ・データ・ブロックに構成して、キャッシュ・メモリにテクスチャAを含むブロックをダウンロードすることもできる。次に、ホスト・コンピュータは、テクスチャAと同じブロックの異なるサブテクスチャ内で主メモリにテクスチャBを構成することができる。このような形態では、ホスト・コンピュータは、(図2の) テクスチャ・マッピング・チップ46の動作を停止させるコマンドを発し、(同じブロック内のテクスチャAおよびBを含む) 新しく構成されたテクスチャ・データ・ブロックをテクスチャ・マッピング・システムのキャッシュ・メモリへダウンロードする。理解されることであろうが、停止状態が実行されず、新たに構成されたデータが主メモリからテクスチャ・マッピング・システムのキャッシュ・メモリにダウンロードされなかったならば、間違ったテクスチャ・マッピング・データが、プリミティブBのレンダリングの間アクセスされる可能性がある。なぜならば、プリミティブBをレンダリングする時、テクスチャBを含むデータ・ブロックに関する読取りキャッシュ・タグがテクスチャAを記憶するキャッシュのデータ・ブロックに対応するブロック・タグと一致するためキャッシュ・ディクトリのヒットが発生するからである。しかし、キャッシュのデータ・ブロックは、テクスチャAに関するテクスチャ・データだけを記憶していてテクスチャBに関するものは記憶していない。

【0142】IX. 3次元プリミティブ・パイプラインのバイパスおよびテクスチャ・マップのダウンロードに関する割り込み方式

上述のように、本発明の1つの機能によって、新しいテクスチャのためのMIPマップは、3Dプリミティブ・データを扱うパイプラインとは別個のデータ経路を経由してテクスチャ・マッピング・ハードウェアのローカル・メモリにダウンロードされる。(図2の) テクスチャ・マッピング基板12および(図4の) テクスチャ・マッピング・チップ46は各々、3Dプリミティブ・データおよびテクスチャ・データをそれぞれ受け取る独立したポートを有する。3Dプリミティブ・データはバス18經由で集線器チップ36から受け取られ、一方、テクスチャ・データは2D加速器チップ34からバス24を經由で受け取られる。従って、新しいテクスチャ・データがホスト・コンピュータ15からテクスチャ・マッピ

ング・チップ 46 へダウンロードされる時、フロントエンド基板 10 を通過する 3D プリミティブ・パイプラインおよびテクスチャ・マッピング・チップ 46 はフラッシュされる必要はなく、このため、新しいテクスチャ・データがホスト・コンピュータからテクスチャ・マッピング・チップへダウンロードされる時は必ず 3D プリミティブ・パイプラインのフラッシュを必要とする従来技術のテクスチャ・マッピング・システムに比較して帯域幅が増大する。

【0143】3D プリミティブ・パイプラインをバイパスしてテクスチャ・データをダウンロードする独立したデータ経路は、テクスチャ・マッピング基板上のローカル・メモリがキャッシュとして実施される本発明の上述の実施形態と相まって特に有効である。上述のように、新しいテクスチャ・データがキャッシュにダウンロードされる場合、テクスチャに関する一連の MIP マップ全体ではなく、必要とされる MIP マップ部分だけがダウンロードされる。このようにして、3D パイプラインのバイパスは、パイプラインをフラッシュすることなくキャッシュ・ミスの処理を取り扱うことを可能にする。

【0144】上述のように、図 3 に示される本発明の 1 つの実施形態において、グラフィックス・システムの特定期間が反復複製して構成されることによって、システムの帯域幅が増加される。テクスチャ・マッピング基板 12 は、2 つのテクスチャ・マッピング・チップ 46 A ならびに 46 B、および 2 つのキャッシュ・メモリ 48 A ならびに 48 B を備えている。この実施形態では、典型的には 2 つのテクスチャ・マッピング・チップの両方が同時に同じテクスチャ・データを使用するプリミティブに関して処理を行うので、両方のキャッシュ・メモリ 48 は同じテクスチャ・データを常に保持する。従って、ある一方でキャッシュ・ミスが発生すれば必ず両方を更新するので、同じテクスチャ・データが別々の動作で両方のキャッシュへダウンロードされる必要のないことが保証されるため、この実施形態はシステムの帯域幅を節約する。

【0145】図 3 の複式テクスチャ・マッピング・チップ実施形態において、各キャッシュ・メモリは、ホスト・コンピュータからダウンロードされるテクスチャ・データのみについて更新され、テクスチャ・マッピング・ハードウェアからローカルに書き込まれることはない。従って、キャッシュの 1 つにおけるキャッシュ・ミスにตอบสนองしてテクスチャ・データがホスト・コンピュータからダウンロードされる時は必ず両方のキャッシュが新しいテクスチャ・データで更新されることが保証されるため、2 つのキャッシュ・メモリの間の整合性が維持される。テクスチャ・マッピング・チップ 46 の 1 つにキャッシュ・ミスが発生し割り込みが生成されると、ダウンロードされたテクスチャ・データを用いて両方のキャッシュ・メモリを更新することができるよう、両方のテ

クスチャ・マッピング・チップ 46 が停止させられる。このように、いずれかのテクスチャ・マッピング・チップから発せられるキャッシュ・ミス信号にตอบสนองして、テクスチャ・マッピング・チップの各々は動作を停止する。更に、本発明は、異なるキャッシュ・ブロックに対する 2 つのテクスチャ・マッピング・チップ 46 の同時キャッシュ・ミスをサポートし、キャッシュ・ミスにตอบสนองして、新しいテクスチャ・データ・ブロックの両方を両方のキャッシュにダウンロードする。

10 【0146】図 2 に示されるように、3D プリミティブ・パイプラインのバイパスは、2D 加速器チップ 34 を通過する 2D プリミティブ・パイプラインを使用してテクスチャ・データをダウンロードすることによって達成される。テクスチャ・マッピング・チップ 46 へテクスチャ・データをダウンロードするデータ経路は、3D プリミティブ・パイプラインをバイパスする点は同じとしても、多数の形態で実施できることは理解されるべきであろう。例えば、ホスト・コンピュータからテクスチャ・マッピング基板へテクスチャ・データをダウンロード

20 する 1 つの専用データ経路を備えることも可能である。
【0147】本発明のグラフィックス・システムのホスト・コンピュータが、同時に動作する複数のプロセスを持ち、プロセスが割り込まれないように一定のシステム資源をロックすることを可能にする何らかの方式を提供する UNIX のようなオペレーティング・システムを使うこともある。ロッキング方式の使用によって、特定のハードウェア資源を使用するプロセスは、それらの資源のロックをはずすまでプロセスはスワップアウトされないことを保証することができる。

30 【0148】本発明の 1 つの実施形態において、急速ロック (fast lock) と遅速ロック (slowlock) という 2 つのタイプのロックがプロセスによる使用のため用意される。急速ロックが使われると、スワップインされるプロセスは、適切なハードウェア資源を検査して、そのプロセスがそれらの資源を使用する最後のプロセスであったか否かを判断する。そうであれば、プロセスはハードウェア資源状態を復元することなく継続する。しかし、最後のものではない場合、遅速ロックが要求され、そのプロセスが最後にスワップアウトされた時の状態にハードウェア資源が復元される。同様の結果を達成することができ

40 多数の代替方法があることは理解されるべきであろう。
【0149】3D プリミティブをレンダリングする間 2D プリミティブ・パイプラインを使用してテクスチャ・データをダウンロードする本発明の実施形態において、2D および 3D プロセスは同時に動かされない。ホスト・コンピュータのオペレーティング・システムによって提供されるロッキング方式の使用によって、3D パイプラインが空でない限り 2D プロセスは開始しないこと、
50 および、2D パイプラインが空でない限り 3D プロセス

は開始しないことを保証することによって、上記制約が守られる。3Dプロセスが開始する時それはロックをかけ、先行プロセスが2Dであった場合2Dパイプラインが空になるまで開始せず待機する。同様に、2Dプロセスが開始する時それはロックをかけ、先行プロセスが3Dであった場合3Dパイプラインが空になるまで開始せず待機する。

【0150】プロセスによって、3Dおよび2D動作の両方を実行し、かつ、遅速ロックを放棄することなく3Dプリミティブと2Dプリミティブの間の切り換えを行うこともできる。そのようなプロセスは、また、3Dパイプラインがハードウェアへ2Dプリミティブ・データをダウンロードする前に空であることを確認し、同様に2Dパイプラインがハードウェアへ3Dプリミティブ・データをダウンロードする前に空であることを確認する方式を実行する。この方式を達成するため、2Dおよび3Dプリミティブ・パイプラインの各々が空であるか否かを標示するレジスタ状態ビットが用意されることもある。2Dおよび3Dプリミティブ・データを使うプロセスは、2Dおよび3Dプリミティブ・データの間の切り換えを行う前にパイプラインが空であることを確認するため、この状態レジスタを読み取る。

【0151】上記の本発明の実施形態はキャッシュとして実施されるテクスチャ・マッピング基板上のローカル・メモリを含むが、本発明はそれに限定されない点は理解されるべきであろう。テクスチャ・マッピング基板上のローカル・メモリがキャッシュでなく、プリミティブがレンダリングされる時テクスチャ・マッピング・データがローカル・メモリで使用可能であるようにするため、3Dプリミティブ・パイプラインとは別の経路を経由して、プリミティブがレンダリングされる前に、プリミティブをレンダリングするために必要とされるテクスチャ・マッピング・データの各ブロックがダウンロードされることを保証するようなその他の技術を使用するテクスチャ・マッピング・システムを実施することもできる。

【0152】更に、ホスト・コンピュータによるローカル・メモリのデータ・ブロックの更新のための割り込みを生成する本発明の方式は、多くの他のアプリケーションについて実施することができ、テクスチャ・マッピング・ハードウェア・システムにおける使用に限定されない点は理解されなければならない。この方式は、処理されるべきデータ・ブロックを記憶する主メモリを備えるホスト・コンピュータ、および処理されるべきデータ・ブロックを記憶するローカル・メモリを備えるデータ処理ハードウェアを含むデータ処理システムにとって利点がある。

【0153】X. キャッシュ・ブロック置き換え方式

上述のように、キャッシュにないテクスチャ・データ・ブロックについてキャッシュ・ミスが発生すると、ホス

ト・コンピュータは、要求されたテクスチャ・データのブロックを(図2の)キャッシュ48へダウンロードする。キャッシュがいっぱいの時キャッシュ・ミスが発生すると、キャッシュ・ブロックの1つが、新しくダウンロードされるテクスチャ・データ・ブロックと置き換えられる。本発明の1つの実施形態において、最も以前に使用されたキャッシュ・ブロックが判定され、キャッシュの活動的ブロックを維持するための置き換え用としてそのブロックが選択される。ホスト・コンピュータ15のメモリ17に記憶されホスト・コンピュータのプロセッサ19上で動くソフトウェア・ルーチンによって、置き換えられるべきキャッシュ・ブロックが決定される。テクスチャ・マッピング・チップ46は、置き換えられるべきキャッシュ・ブロックを決定するソフトウェア・ルーチンをサポートする2組のレジスタを含む。キャッシュ・ミスが発生すると、これらのレジスタが、3Dバイパス・データ経路を経由してホスト・コンピュータによって読み取られ、置き換えられるべきキャッシュ・ブロックを決定する際に使用される。

【0154】レジスタの第1の組は、バンク0およびキャッシュ48の1つにそれぞれ対応するように配置され、最も最近使用された2つの32ビット・レジスタMRU0およびMRU1(集合的にMRU、Most Recently Usedと呼ぶ)を含む。これらのレジスタの各ビットは、その対応するキャッシュ・バンク内に含まれる32個のキャッシュ・ブロックの1つに対応する。あるブロックについてキャッシュ・ヒットが発生するたびに、最も最近使用されたレジスタがキャッシュ・ヒットを蓄積するように、MRU0またはMRU1における対応するビットがセットされる。

【0155】レジスタの第2の組は、バンク0およびキャッシュの1つにそれぞれ対応するように配置され、現在使用中の32ビット・レジスタCU0およびCU1(集合的にCU、Currently Usedと呼ぶ)を含む。CU0またはCU1の1つのビットがセットされている場合、それは、対応するキャッシュ・ブロックがキャッシュのミニディレクトリに現在存在し、従って置き換えられるべきものではないことを標示する。キャッシュのミニディレクトリの詳細は後述する。

【0156】キャッシュ・ミスが発生し、ホスト・コンピュータへの割り込みが起きると、図19の流れ図によって示されるソフトウェア・ルーチンが、ホスト・コンピュータのプロセッサ19によって実行され、ダウンロードされるように要求されたテクスチャ・データを含むブロックとどのキャッシュ・ブロックを置き換えるべきかが決定される。ソフトウェア・ルーチンは、置き換えルーチンを実行する際に使用される2つの64ビット状態ワード(すなわちBLOCKS_TO_USEおよびBLOCKS_BUSY)を保持する。これらの状態ワードの64状態ビットの各々は、64個のキャッシュ・

ブロックの1つに対応する。

【0157】図19のステップ300において、それぞれ初期的に置き換え可能状態にあることを各ビットが標示するようにBLOCKS__TO__USEが初期化される、ステップ302において、ルーチンはキャッシュ・ミス割り込みが受け取られたか否かを判断するため継続的に検査し、割り込みが検出されると、ステップ304へ進み、3Dバイパス・データ経路を経由してレジスタMRUおよびCUを読み取る。上述のように、2つのテクスチャ・マッピング・チップが使われている本発明の実施形態においては、2つのチップのキャッシュ・メモリは、同じテクスチャ・データを常時保持する。従って、システムが2つのテクスチャ・マッピング・チップ46を含むならば、両方のチップのレジスタMRUおよびCUが読み取られ、ルーチンは、置き換え用として、いずれかのテクスチャ・マッピング・チップにおいて最も以前に使用されたキャッシュを選択することができ。ステップ306において、MRUまたはCUでオンにされているビットに対応するBLOCKS__TO__USEのビットをオフにする。2つ以上のテクスチャ・マッピング・チップが使用される実施形態では、MRUとCUの論理和を使用して、オフにすべきBLOCKS__TO__USEのビットを決定する。

【0158】ステップ308で、BLOCKS__TO__USEのいずれかのビットがオンにされているかという判断が行われ、少なくとも1つがオンであれば、ルーチンはステップ310へ進み、BLOCKS__TO__USEのオンにされているビットの数が所定のしきい値より少ないか否かの判断が行われる。このステップは、複数のキャッシュ・ミスに関するキャッシュ・ブロック使用実績の維持を援助し、(後述される)将来のキャッシュ・ミス割り込みの適切な処理を確実にするため実行される。BLOCKS__TO__USEのオンにされているビットの数が所定のしきい値より少ない場合、ルーチンはステップ312へ進み、MRUのビットのすべてがオフにされる。この結果、MRUは、現在処理されているキャッシュ・ミスの後に発生するキャッシュ・ヒットについてのみキャッシュ・ヒットを累積し始める。本発明の1つの実施形態では、上記しきい値は、BLOCKS__TO__USE中オンにされた11ビットと設定され、これは11個のキャッシュ・ブロックが置き換えに使用できることを標示する。

【0159】ステップ312でMRUがクリアされた後、または、ステップ310でBLOCKS__TO__USEのオンのビット数が所定のしきい値未満であると判断された後、ルーチンはステップ314へ進み、BLOCKS__TO__USEでオンにセットされたビットの1つが、ダウンロードされるべきテクスチャ・データの新しいブロックとの置き換え用として選択される。ステップ314において置き換えのため選択されたブロック

が、図21を参照して後に説明する方法で、テクスチャ・データの新しいブロックによって置き換えられる。ステップ314において、置き換えられるブロックが選択された後、ルーチンはステップ302へ戻り、別のキャッシュ・ミス割り込みを待つ。

【0160】ステップ308でBLOCKS__TO__USEのビットにオンにされているビットがないと判断されると、ルーチンはステップ316へ進み、BLOCKS__BUSYが、MRUとCUの論理和と等しく設定される。従って、BLOCKS__BUSYでセットされているビットだけが、MRUまたはCUレジスタのどちらかにセットされているビットに対応する。その後、BLOCKS__TO__USEは、BLOCKS__BUSYの補数に等しく設定される。このような形態では、MRUおよびCUでオンにされ、置き換え用に選択されるべきでないことを標示するビットに対応するビットを除いて、BLOCKS__TO__USEの各ビットはオンにされる。

【0161】ステップ316でBLOCKS__TO__USEがBLOCKS__BUSYの補数と等しく設定された後、ルーチンはステップ318へ進み、BLOCKS__TO__USEのいずれかのビットがオンになっているか判断される。BLOCKS__TO__USEの少なくとも1つのビットがオンとなっていれば、ルーチンはステップ310ないしステップ314へ進み、BLOCKS__TO__USEのオンにされたビットの数がしきい値を下まわっていればMRUのすべてのビットがオフにされ、BLOCKS__TO__USEのオンにされたビットの1つが置き換えのため上述の方法で選択される。

【0162】オンとなっているビットがBLOCKS__TO__USEに1つもない場合、ルーチンはステップ320へ進み、3つのアクションが取られる。第1に、BLOCKS__TO__USEのオンにされたビットの数が所定のしきい値より少ないのでMRUのすべてのビットをオフにする。第2に、BLOCKS__BUSYは、CUレジスタと等しい値に設定される。上述のように、各CUレジスタは、対応するキャッシュ・ミニディレクトリに現在維持されているキャッシュ・ブロックを示しているので、置き換えられてはならない。複数のテクスチャ・マッピング・チップが使われている場合、BLOCKS__BUSYは、CUレジスタの論理和と等しく設定される。最後に、BLOCK__TO__USEが、BLOCKS__BUSYの補数に等しく設定される。その結果、テクスチャ・マッピング・チップの1つのキャッシュ・ミニディレクトリに現在維持されているデータ・ブロックのビットに対応するビットを除いて、BLOCKS__TO__USEの各ビットはオンにされる。ルーチンはステップ314へ進み、BLOCKS__TO__USEの中のオンにされたビットの1つが置き換え用として選択される。このように、ミニディレクトリにあるもの

外のキャッシュのブロックのいずれかを置き換え用として選択することができる。

【0163】図19に示される本発明の実施形態は、キャッシュ・ミスが発生する時LRU法（すなわち最も以前に使用されたものを対象とする方式）を用いてキャッシュ・ブロックを置き換える置き換え方式を使う。本発明の有効範囲を逸脱することなくこの方式に種々の変更を加えることができることは理解されなければならない。例えば、図19に示される実施形態において、MRUハードウェア・レジスタが、複数のキャッシュ・ミスが潜在的に含まれる可能性のある一定時間にわたってキャッシュ・ヒットを収集するために使用され、BLOCKS_TO_USEのオンにされたビットの数が所定のしきい値を下回った時にのみMRUレジスタをクリアする。更には、ソフトウェア状態ワードのBLOCKS_BUSYは、BLOCKS_TO_USEのすべてのビットがオフであることが判明した場合にのみステップ316またはステップ320で更新される。代わりの方法として、キャッシュ・ミス割り込みが受け取られる度毎にMRUを使用してBLOCKS_BUSYを更新することによって置き換えを実行することができる。この形態で、複数のキャッシュ・ミスが潜在的に含まれる可能性のある一定時間にわたってキャッシュ・ヒットの実績を累積するためソフトウェア状態ワードBLOCKS_BUSYを使用することができ、そして、ハードウェア・レジスタMRUを、ミスとミスの間のヒットを累積するため使用することができる。

【0164】更に、MRUをクリアする効果を持つBLOCKS_TO_USEのオン・ビットのしきい値が上述の実施形態において11ブロックについてオンにセットされているというものであったが、この数値は明らかに変更することができることは理解されるべきであろう。このしきい値は、ルーチンが、ステップ308において、BLOCKS_TO_USEのビットがどれもオンになっていない状態に遭遇する回数に影響を及ぼす。この状態は最も最近使用されたキャッシュ・ブロックについて（ステップ316またはステップ320で）BLOCKS_TO_USEの更新につながるものでこの状態を回避することが望ましい。BLOCKS_TO_USEにおいてオンにセットされるビットが、複数のキャッシュ・ミスの処理を通して使用されたことのないブロックを反映するように、高分解能を備えることが望ましい。従って、MRUクリアにつながるBLOCKS_TO_USEのオン・ビットのしきい値を制御することによって、ステップ308においてBLOCKS_TO_USEのオン・ビット数を判断する本ルーチンの回数が最小限に抑えられ、最も以前に使用されたキャッシュ・ブロックを決定する場合の望ましい分解能が与えられる。

【0165】ホスト・コンピュータ上で実行されるソフトウェア・ルーチンによって実施される上述の置き換え

方式がキャッシュ・メモリに関する使用に限定されるものではないことは理解されるべきであろう。ローカル・メモリが、処理されるデータ・ブロックを含み、追加データ・ブロックがホスト・コンピュータからローカル・メモリへダウンロードされる時にローカル・メモリ内のデータ・ブロックが置き換えられるようなデータ処理システムのどのようなシステムにおいても上記置き換えルーチンを使用することが可能である。

【0166】XI. キャッシュ動作の実行禁止

- 10 本発明の1つの実施形態において、プリミティブのレンダリングの間テクスチャ・データが必要とされる前に任意の3Dプリミティブに関するテクスチャ・データがメモリ48にダウンロードされるようにするため、キャッシュ・ミスを実行禁止(disable、ディスエーブル)にすることによってテクスチャ・マッピング基板上のローカル・メモリ48のキャッシュ動作を実行禁止にする機能が備えられる。テクスチャ・マッピング・チップ46の各々は、キャッシュとしてのローカル・メモリの動作が実行可能状態であることを標示する状態ビットを含む。
- 20 この状態ビットがオンになっていれば、キャッシュ・ミスの発生が、ホスト・コンピュータの割り込みおよびテクスチャ・マッピング・チップの停止を引き起こす。しかし、この状態ビットがオフとなっていれば、テクスチャ・マッピング基板のローカル・メモリ48はキャッシュとして機能せず、いかなるプリミティブに対するテクスチャ・データも、キャッシュ・ミスが発生しないように、該プリミティブによって必要とされる前にメモリ48にダウンロードされる。本発明の1つの実施形態において、キャッシュとしてのローカル・メモリの動作が実行禁止にされ、テクスチャ・データが、テクスチャ・マッピング基板上のローカル・メモリに3Dプリミティブ・パイプラインを経由してダウンロードされ、テクスチャ・データと対応する3Dプリミティブ・データの同期が取られる。

【0167】XII. キャッシュ・ミスに回答してテクスチャ・データをダウンロードする方式を支援するテクセル・ポート・レジスタ

- 上述のように、(図2の)テクスチャ・マッピング・チップ46は、ホスト・コンピュータ15からダウンロードされるテクスチャ・データを受け取るため使用される(図4の)テクセル・ポート92を含む。テクセル・ポートは、テクセル・データのダウンロードを支援する多数のレジスタを含む。それらのレジスタのいくつかは、上述のレジスタMRUとCUを含む。その他のテクセル・ポート・レジスタには、コマンド・レジスタ、状態レジスタ、テクセル・データ・レジスタ、ディレクトリ・タグ・レジスタ、キャッシュ・アドレス・レジスタおよびパイプ・タグ・レジスタが含まれる。それぞれの機能を以下に説明する。

- 50 【0168】3Dプリミティブ・パイプライン経由でこ

れらレジスタへの書き込みを行えるように、テクセル・ポート・レジスタへのアクセスが提供される。3Dパイプラインが使用中の時でも、単にパイプラインに置かれているレジスタ書き込み用データを用いて、テクセル・ポート・レジスタに書き込むことができる。さらに、テクセル・ポート・レジスタは、24ビット・バス24

(図2)上に備わる3Dパイプライン・バイパスを経由してアクセスすることも可能である。テクセル・ポート・レジスタをアクセスする場合、バス24の8ビットが、どのテクセル・ポート・レジスタの読み取りまたは書き込みを行うべきかを指定するレジスタ・アドレスとして使用され、データがテクセル・ポート・レジスタに書き込まれる時、上記バス24のその他の16ビットがデータを提供する。

【0169】テクセル・ポート・レジスタの構成は図20に示されている通りである。本発明の1つの実施形態において、テクセル・ポート・レジスタの各々のビット数は32ビットである(ただし一部のレジスタの多数のビットは未使用)。

【0170】A. テクセル・コマンド・レジスタ

テクセル・コマンド・レジスタは、詳細は後述するが、キャッシュ・ミスを取り扱うホスト・コンピュータのソフトウェア・ルーチンによって使用される多数のビットを含む。停止ビット350は、ソフトウェア割り込み処理ルーチンによってセットされ、テクスチャ・マッピング・チップに対してその動作を停止するように指示する。上述のように、2つのテクスチャ・マッピング・チップが備わる本発明の実施形態においては、両方のキャッシュが整合性を保つようにどちらのキャッシュ・ミスが発生しても両方のテクスチャ・マッピング・チップは同じテクスチャ・データで更新される。どちらかのテクスチャ・マッピング・チップのキャッシュ・ミスが受け取られると、それぞれのテクセル・コマンド・レジスタの停止ビット350をセットすることによって両方のテクスチャ・マッピング・チップが停止させられる。キャッシュ・ミスにตอบสนองして新しいテクスチャ・データがホスト・コンピュータからダウンロードされた後、キャッシュ・ミスを扱うソフトウェア・ルーチンがコマンド・レジスタに書き込みを行うことによって停止ビットがクリアされる。

【0171】割り込み可能ビット352は、それがオンの時、キャッシュ・ミス発生の際テクセル・ポートからの割り込みを可能にする。このビットは、テクスチャ・マッピング基板12(図2)上のローカル・メモリ48をキャッシュとして動作させない上述の機能を提供する場合にはオフにセットされる。

【0172】書き込みLocki0354ならびに書き込みLocki1ビット356は、テクセル・ポート・レジスタに関する書き込み可能/禁止を制御する。Lockiは、テクスチャ・マッピング・チップ46を識別するために使

用される短縮形名称である。2つのチップが使用される本発明の実施形態においては、それら2つのチップはLocki0ならびにLocki1と呼ばれる。単一のテクスチャ・マッピング・チップが使われ時は、そのチップはLocki0として識別される。テクセル・ポート・レジスタのいずれかに対する書き込みを行うコマンドがバス24経由で受け取られると、各テクスチャ・マッピング・チップ(すなわちLocki0ならびにLocki1)はその書き込みビットが使用可能とされているかを判断するためそのコマンド・レジスタを検査し、使用可能であれば、受け取られた書き込みコマンドに従ってそのテクセル・ポート・レジスタを更新する。従って、書き込みLocki0354ならびに書き込みLocki1ビット356の値を制御することによって、ホスト・コンピュータ上で動くソフトウェア・ルーチンが、2つのテクスチャ・マッピング・チップのテクセル・ポート・レジスタに、別々に、あるいはまた同時に書き込むことができる。

【0173】Locki読取りビット358は、テクスチャ・マッピング・チップの1つのテクセル・ポート・レジスタの読取りを可能にする。テクセル・ポート・レジスタを読み取るコマンドがテクセル・バス24経由で受け取られると、一時点において、テクスチャ・マッピング・チップ(複数)の中の1つだけが応答して、バス上へそのテクセル・ポート・レジスタの内容を送出する。2つのテクスチャ・マッピング・チップが備わる実施形態において、各々は、該チップがLocki0かLocki1のどちらかであることを示すハードウェア・ピンを備える場合がある。Locki読取りビットがソフトウェアによってセットされる場合、それはLocki1からの読取りが可能とされていることを示し、Locki読取りビットがセットされていない場合、それはLocki0からの読取りが可能とされていることを示す。テクセル・コマンド・レジスタの形式が、両方のテクスチャ・マッピング・チップ(Locki0とLocki1)に同じデータで同時に書き込むことを可能にするため、レジスタへの書き込みのため1回の書き込みサイクルだけでよい点は前述の記載から理解されることであろう。

【0174】B. テクセル状態レジスタ

テクセル・ポート状態レジスタは、オンにセットされる時、システムが2つのテクスチャ・マッピング・チップを含むことを示す複数Lockiビット360を含む。コマンド・レジスタのビット352がオンの時は必ず割り込み可能ビット362はオンにされ、テクスチャ・マッピング・チップのローカル・メモリはキャッシュとして機能し、キャッシュにないテクスチャ・データが必要とされる時、ホスト・コンピュータに割り込むためのキャッシュ・ミスを生成することを標示する。このビットは、コマンド・レジスタと共に状態レジスタにも含められるので、状態レジスタを単に読むだけでテクセル・ポートの状態を読み取ることができる。

【0175】割り込みがテクスチャ・マッピング・チップから発生し、該チップが新たなテクスチャ・データがダウンロードされるのを待つ時、割り込み有効ビット364がオンにされる。パイプ・タグ・レジスタ（後述）に記憶されているキャッシュ・ミスが発生したキャッシュ読取りタグと一致するキャッシュ・タグを用いてキャッシュ・ディレクトリ・タグ・レジスタ（後述）に書き込みが行われると、このビットはクリアされる。

【0176】状態レジスタは、キャッシュ・ミスが発生する時テクスチャ・マッピングの停止を支援する2つのビットを含む。停止可能ビット368は、コマンド・レジスタの停止ビット350がセットまたはクリアされる毎に、ホスト・コンピュータ上のソフトウェア・ルーチンによってそれぞれセットまたはクリアされ、当ビットがオンの時動作を停止するようにテクスチャ・マッピング・チップに指示する。このビットがコマンド・レジスタと共に状態レジスタに備えられるので、テクスチャ・マッピング・チップの状態が単一のレジスタに記憶される。キャッシュ・ミスが発生し、キャッシュ・ディレクトリが新たなデータがダウンロードされるのを待つ時、割り込み有効ビット364がテクスチャ・マッピング・チップのハードウェアによってオンにされる。キャッシュ・ミスが発生したブロック・タグと一致するキャッシュ・タグを用いてキャッシュ・ディレクトリ・タグ・レジスタ（後述）に書き込みが行われると、このビットはクリアされる。

【0177】C. パイプ・タグ・レジスタ

パイプ・タグ・レジスタは、テクスチャ・マッピング・チップのパイプラインによってインデックス付けされた最後のブロック・タグを記憶する。キャッシュ・ミスが発生すると、パイプ・タグ・レジスタは、キャッシュ・ミスが発生したブロック・タグ370を記憶する。テクセル・ポート・バス24を経由してパイプ・タグ・レジスタを読み取ることによって、キャッシュ・ミス割り込みに応答するソフトウェアが、キャッシュ・ミスに応答してダウンロードされなければならないキャッシュ・ブロックのタグを決定することができる。

【0178】D. テクセル・データ・レジスタ

テクセル・データレジスタは、キャッシュ・ミスが発生した時テクスチャ・データをキャッシュ48へダウンロードするために使用される。上述のように、各テクセルは、 α を示す1バイト372、赤の値を表す1バイト374、緑の値を表す1バイト376および青の値を表す1バイト378を含む32ビットのデータによって表される。

【0179】E. テクセル・キャッシュ・アドレス・レジスタ

テクセル・キャッシュ・アドレス・レジスタは、キャッシュへテクセル・データを書き込み、ブロック・タグをキャッシュ・ディレクトリに書き込むために使用され

る。上述のように、キャッシュは、各ブロックが256×256アレイのテクセルを含む64ブロックのテクスチャ・データを記憶する。テクセル・キャッシュ・アドレス・レジスタは、キャッシュにおける64ブロックのうちで読み込みまたは書き込みの対象となる特定の1つのブロックを識別する6ビットのブロック・インデックス・フィールド380を含む。加えて、このレジスタは、上記ブロック・インデックス・フィールドで識別されるブロック内で読み込みまたは書き込みの対象となる特定のテクセル・アドレスを識別する16ビットのブロック・アドレス・フィールド382を含む。キャッシュ・ミスに応答してテクスチャ・メモリにデータがダウンロードされる時、ブロック・インデックスが、上述のLRU（すなわち最も以前に使用されたものを選択する）置き換え法を使用してソフトウェア・ルーチンによってセットされ、ブロック・アドレス・フィールド382が最初のテクセルをブロックに書き込むためゼロに初期化される。キャッシュ・アドレス・レジスタは、テクセル・データ・レジスタがアクセスされると必ずブロック・アドレス・フィールド382を自動的に増分する。このように、キャッシュ・ブロック内のブロック・アドレスのすべてにわたってブロック・アドレス・フィールドが増分され、テクセル・データの新しいブロックがキャッシュに書き込まれる。

【0180】F. テクセル・ディレクトリ・タグ・レジスタ

テクセル・ディレクトリ・タグ・レジスタは、キャッシュ・ブロック・タグを表す23ビットのブロック・タグ・フィールド384を含み、キャッシュ・アドレス・レジスタのブロック・インデックス・フィールド380によって定義されるキャッシュ・ディレクトリ・エントリを書き込むために使用される。上述のように、キャッシュ・ブロック・タグの23ビットは、8ビットのテクスチャID、7ビットのS座標、7ビットのT座標、および、ブロック・タグに対応するテクスチャ・データのブロックによって表されるマップのマップ番号を識別する付加ビット（1ビット）を表す。キャッシュ・ミスに応答してテクスチャ・データの新しいブロックがホスト・コンピュータからダウンロードされる時、そのブロック・タグは、テクセル・バス24経由でディレクトリ・タグ・レジスタにロードされる。ディレクトリ・タグ・レジスタから、ブロック・タグは、キャッシュ・アドレス・レジスタのブロック・インデックス・フィールド380によって識別されるキャッシュ・ディレクトリ・エントリに書き込まれる。上述のように、キャッシュ・ミスを発生させたパイプ・タグ・レジスタのタグと一致するディレクトリ・タグ・レジスタにブロック・タグが書き込まれると、キャッシュ・ミス割り込みはクリアされる。

【0181】XIII. キャッシュ・ミス割り込みを取り扱

ソフトウェア・ルーチン

上述の記載から理解されるように、テクセル・ポート・レジスタは、キャッシュ・ミス割り込みに応じて必要なテクスチャ・データをダウンロードする機能を遂行するホスト・コンピュータ15上のソフトウェア・ルーチンによって使用される。図21は、このソフトウェア・ルーチンの流れ図を示す。ステップ400において、`Locki0`および`Locki1`両方のテクセル・コマンド・レジスタにおける停止ビット350がセットされる。次にルーチンはステップ402へ進み、テクセル状態レジスタの停止ビット368を読み、両方の`Locki`が停止したか否か判断する。`Locki0`および`Locki1`が停止したことが確認できるまでルーチンは両方の状態レジスタの読み取りを継続する。両方が停止したと確認すると、ルーチンはステップ404へ進む。システムがただ1つのテクスチャ・マッピング・チップ46（すなわち、`Locki0`）を含む場合、`Locki0`は、テクセル・バス24上に`Locki0`のテクセル・ポート・レジスタの内容を送出することによって`Locki1`のテクセル・ポート・レジスタを読み取る要求に応答する。このように、ソフトウェア・ルーチンがステップ402で両方の`Locki`が停止したか否か判断する時、`Locki0`が、`Locki0`が停止した場合のように`Locki1`の読み取り要求に応答するので、処理はステップ404へ進む。

【0182】ステップ404で、`Locki0`がキャッシュ・ミスによって割り込みを起こした否かを調べるため`Locki0`のテクセル状態レジスタ中の割り込み有効ビット364を読み込まれる。もし割り込みが発生していれば、ルーチンはステップ406へ進み、`Locki0`のパイプ・タグ・レジスタが読み取られ、キャッシュ・ミスを発生させたテクスチャ・データのブロックのブロック・タグを識別する。ソフトウェア・ルーチンはこのブロック・タグを使用して、ホスト・コンピュータのメモリ17（図2）に記憶されているテクスチャ・データの対応するブロックにアクセスし、ステップ408で、キャッシュのどのブロックを、ダウンロードされるテクスチャ・データの新しいブロックと置き換えるべきかを決定する。この決定は、図19に関連して上述したLRU法を使用して実行される。

【0183】上述のように、システムが2つのテクスチャ・マッピング・チップを含む場合、各々におけるキャッシュは全く同じエントリを持つように維持される。従って、テクスチャ・マッピング・チップの1つに発生したキャッシュ・ミスに応答してホスト・コンピュータからダウンロードされるテクスチャ・データは、両方のチップのキャッシュに書き込まれる。かくして、置き換えられるキャッシュ・ブロックが識別されたならば、ルーチンはステップ410へ進み、`Locki0`および（`Locki1`が存在する場合）`Locki1`におけるキャッシュ

・アドレス・レジスタが、ステップ408で決定されたブロック・インデックスを用いて書き込まれる。ステップ412で、キャッシュ・ミスに応答してテクスチャ・キャッシュにダウンロードされるべきテクスチャ・データのブロックのブロック・タグを用いてディレクトリ・タグ・レジスタに書き込みが行われ、ステップ414において、テクスチャ・データがテクセル・データ・レジスタに書き込まれる。このように、本ルーチンは、キャッシュ・ミスのあったテクスチャ・データのブロックをダウンロードしてキャッシュにこのデータ・ブロックを書くことによってキャッシュ・ミスに応答する。

【0184】ステップ406ないしステップ414においてテクスチャ・データのブロックが`Locki0`および`Locki1`にダウンロードされたあと、あるいは、ステップ404で`Locki0`が割り込まれなかったと判断されたなら、ルーチンはステップ416へ進み、キャッシュ・ミスが発生したのが`Locki1`であることを示す`Locki1`状態レジスタの割り込み有効ビット364を調べる。上述のように、システムが1つのテクスチャ・マッピング・チップだけを含む場合、`Locki0`が`Locki1`テクセル・ポート・レジスタの読み取りに応答する。`Locki0`が`Locki1`の状態レジスタの読み取りに応答する場合、その割り込み有効ビット364はマスクされているので、ソフトウェア・ルーチンはステップ416において`Locki1`が割り込みを起こしていないと判断する。このマスキングによって、`Locki0`からの割り込みを再プロセスしてテクスチャ・データをダウンロードすることが防止される。従って、ただ1つのテクスチャ・マッピング・チップだけが備わるシステムにおいては、ルーチンはステップ416で`Locki1`に割り込みが発生していないと判断し、ステップ418へ進み、`Locki0`のコマンド・レジスタの停止ビット350がオフにされ、これによって、テクスチャ・マッピング・チップがそのパイプラインにおけるプリミティブ処理を続行することが可能とされる。

【0185】システムが2つのテクスチャ・マッピング・チップを含む場合、ルーチンはステップ416で`Locki1`が割り込みを起こしたか否かを判断し、起こしていなければ、ステップ418へ直接進み、両方のテクスチャ・マッピング・チップの停止ビットをオフにして両方のチップがプリミティブ処理を続行することを可能にする。しかし、ステップ416で、`Locki1`がキャッシュ・ミスに応答して割り込みを起こしたと判断すれば、ルーチンは、ステップ420ないしステップ424へ進み、`Locki0`の割り込みを処理するステップ406ないしステップ414と同様の方法で`Locki1`の割り込みを処理する。次にルーチンはステップ418へ進み両方のテクスチャ・マッピング・チップの停止ビットをオフにする。

【0186】2つのテクスチャ・マッピング・チップを

備えるシステムにおいては、両方のチップが同じブロック・タグについてまたは異なるブロック・タグについて同時にキャッシュ・ミス割り込みを生成することができる点は理解されるべきであろう。両方のテクスチャ・マッピング・チップが、同じブロック・タグについてキャッシュ・ミス割り込みを生成する場合、割り込みは、ステップ400ないしステップ414で処理される。従って、ステップ412においてキャッシュ・ミスのあったブロック・タグを両方のLockiのディレクトリ・タグ・レジスタに書き込むことによってLocki1からの割り込みがクリアされるので、ステップ416においてルーチンはLocki1の割り込みを検出しない。このようにして、図21に示される方法が、いずれのテクスチャ・マッピング・チップからの割り込みに対してもそれぞれ個別的に、あるいは、両方の割り込みに同時に、応答することができる。

【0187】XIV. キャッシュ・ミニディレクトリおよび主ディレクトリ

上述のように、本発明の1つの実施形態において、キャッシュは、1ブロックが256×256テクセル・データからなる64ブロックのテクセル・データ、および23ビットのブロック・タグを持つ64のエントリを含む完全連想型キャッシュ・ディレクトリを含む。本発明が3線形補間モードで動作する場合、8個のテクセル読取りが、ある1つのピクセルについて所望のテクセル・データを決定するために実行される。それら8個のテクセルのうち、ある1つのマップにおける4個のテクセルが1回の読取り動作で同時に読み出され、別のマップの4個のテクセルが第2の1回の読取り動作で同時に読み出される。当該ピクセルがキャッシュ・ブロック境界を隣接するマップの位置に対応する場合、1つのマップ内で所望のテクセル・データを生成するためキャッシュから読み取られる4つのテクセルが各々異なるキャッシュ・ブロックに記憶されていることがある。このように、各ピクセルに関するキャッシュからの4つのテクセルの同時の読取りは、キャッシュ・ディレクトリにおける64ブロック・タグ・エントリと4回の別々の比較を行うことを必要とする場合がある。

【0188】従来技術の完全連想型キャッシュは、次の2つの形態の1つで動作する。第1の形態は、1つの読取りタグを1回のサイクルであらゆるキャッシュ・タグ・エントリと比較することができるように、キャッシュ・タグ・エントリ毎に別々のハードウェア比較器を備えるものである。そのような技術は、4回の読取りを同時に行う本発明において高価なハードウェア・コストの原因となる254（すなわち4×64）個の23ビット比較器を必要とするであろう。従来技術の完全連想型キャッシュによって使われる第2の技術は、1つのキャッシュ・タグ比較器を使用し、各キャッシュ・エントリは、読取りタグと順次比較される。そのような技術は、

1回の読取り動作の間に読み取られる4つのテクセルの各々がキャッシュにあるか否かを判断するためキャッシュ・ディレクトリに対する256回の読取りサイクルが潜在的に必要とされるため、本発明のシステム帯域幅にマイナスの影響を与える。

【0189】これらの問題を克服するため、本発明のキャッシュ・システムは、ミニディレクトリ（図22）および主ディレクトリ（図23）を含む。ミニディレクトリは、完全連想型ディレクトリであって、対応するブロック・インデックスと共に、5つの最も最近読まれたキャッシュ・ブロック・タグを含む。図21に示されるように、ミニディレクトリ500は、ミニディレクトリから出力501-505上にそれぞれ出力される5つのエントリを含む。それらエントリの各々は、4グループのタグ比較器507-510に接続される。タグ比較器507-510の各グループは、5つの23ビット比較器（図示されていない）を含み、双線形または3線形補間が実行される時1回の読取り動作で実行される4つのキャッシュ読取りタグの1つに対応する。このように、完全連想型ミニディレクトリの特徴が、同時に読み取られるタグの数にミニディレクトリのエントリの数を乗じた数に等しい20個の23ビット比較器を用いて実現される。

【0190】同時に読み込まれるある1つのピクセルに関する4つのキャッシュ読取りタグが、該ピクセルが対応するマップ位置に最も近い4つのテクセルを含むキャッシュ・ブロックを識別する。それらの4つのキャッシュ読取りタグは、左上（UL、Upper Left）タグ、右上（UR、Upper Right）タグ、左下（LL、Lower Left）タグおよび右下（LR、Lower Right）タグと呼ばれる。左上、右上、左下および右下のテクセルのためのキャッシュ読取りタグは、それぞれ左上、右上、左下および右下のタグ比較器507-510グループに接続される。タグ比較器507-510の各グループは、その対応するキャッシュ読取りタグをミニディレクトリに記憶されている5つのブロック・タグと比較して、タグがミニディレクトリ・エントリの中の1つと一致することを示すヒット出力を生成し、同時に対応するテクセル・データ・ブロックが記憶されているキャッシュの位置を標示するブロック・インデックスを出力する。

【0191】上述から認められるように、4つのキャッシュ読取りタグ（UL、UR、LL、LR）の各々がミニディレクトリにあれば、対応する4つのテクセル・データ・ブロックが記憶されているキャッシュの位置を標示するブロック・インデックスを決定するため必要とされるディレクトリ・アクセスはただ1回でよい。1つまたは複数の読取りタグがミニディレクトリにない場合にだけ主キャッシュ・ディレクトリに対するアクセスが行われる。ミニディレクトリ500は、ミニディレクトリにおいてキャッシュ読取りタグ・ミスが発生するた

びごとに更新されるので、ミニディレクトリ 500 は常にテクスチャ・データの最も最近アクセスされた 5 つのブロックのブロック・タグを保持する。

【0192】4つのキャッシュ読み取りタグの1つまたは複数がミニディレクトリでヒットしない場合、主キャッシュ・ディレクトリ 520 (図 22) に対するアクセスが行われる。上述のように、主ディレクトリは、各々が 1 つのブロック・タグを持つ 64 のエントリを含む。主ディレクトリは、64 個の 23 ビット比較器 522 を備えるため、1 つのキャッシュ読み取りタグは 1 回のサイクルで主ディレクトリのすべてと比較することができる。比較器 522 は、キャッシュ読み取りタグが主ディレクトリのエントリの 1 つと一致したことを標示する信号を生成し、読み取りタグと一致した比較器の位置を使用して、テクセル・データの対応するブロックがキャッシュに存在する位置を標示するブロック・インデックスを生成する。読み取りタグが主キャッシュ・ディレクトリのどのエントリとも一致しない場合、キャッシュ・ミスが生成され、その結果、ホスト・コンピュータの割り込みが発生し、テクスチャ・データの要求されたブロックが上述の方法でダウンロードされる。

【0193】上述のように、4つのキャッシュ読み取りタグ (UL、UR、LL、LR) の 1 つまたは複数がミニディレクトリをヒットしない場合のみ主キャッシュ・ディレクトリ 520 がアクセスされる。キャッシュ読み取りタグの複数がミニディレクトリと一致しない場合、各キャッシュ読み取りタグ毎に別々のサイクルで主ディレクトリをアクセスしなければならないとすれば、処理能力低下の負荷を減らすことが望ましい。そのような成果を達成するため、本発明の 1 つの実施形態において、図 24 に示されるように、6つの比較器 526-530 が追加される。これら 6つの比較器は、同時にアクセスされる 4つのキャッシュ読み取りタグの各々を一致するものがあるか調べるため他のものと比較する。これらの比較器は、ULタグをURタグと比較する比較器 526、ULタグをLLタグと比較する比較器 527、ULタグをLRタグと比較する比較器 528、URタグをLLタグと比較する比較器 529、URタグをLRタグと比較する比較器 530、およびLLタグをLRタグと比較する比較器 531 を含む。

【0194】比較器 526-532 によって実行される比較は、処理性能の低下が起きないように他の比較と並列的に実行される。例えば、キャッシュ読み取りタグがミニディレクトリと比較されるサイクルの間に、あるいは、ミニディレクトリにおけるミスを起こした最初のキャッシュ読み取りタグが主ディレクトリと比較される時のサイクルの間に、上記の比較を実行することができる。少なくとも 2 つのキャッシュ読み取りタグが主ディレクトリにおいてヒットせず、かつそれらが等しいものであると判断されれば、比較器 526-532 の出力は、これら

少なくとも 2 つのキャッシュ読み取りタグについては主ディレクトリを 1 回だけアクセスすればよいことを示す。このようにして同一のタグについて主ディレクトリをアクセスする場合複数サイクルを伴う必要がなくなるので、複数のキャッシュ読み取りタグがミニディレクトリで一致ミスを起こす場合のシステム帯域幅に対する影響を最小限にとどめることができる。

【0195】上述のことから認められるように、キャッシュ・ミニディレクトリを利用する本発明の実施形態は、高いシステム帯域幅を達成する一方、キャッシュ・ディレクトリを実施するハードウェアの数量を比較的少なくするという対立する目標を効率的に均衡させる。複数のキャッシュ読み取りタグがミニディレクトリで一致しない場合の処理性能の低下はアプリケーションに依存する。それぞれが 4 つのキャッシュ読み取りタグからなるユニークな 2 セットをミニディレクトリに対して 2 サイクル毎に処理することもできるが、典型的には 4 つのキャッシュ読み取りタグの各セットにおいてただ 1 つまたは 2 つのユニークなブロック・タグが現れると考えられる。上述のように、1 つのオブジェクトの複数ピクセルがレンダリングされ、そして、3 線形補間法が使われる場合、隣接するピクセルが、MIP マップの同じ 2 つのマップに頻繁に対応するので、キャッシュに対する読み取りが 2 つのマップを記憶するキャッシュ・ブロックの間で連続的に切り換わることを必要とする。図 22 に示される実施形態において、現在処理されている読み取りタグのセットに対する 4 つのユニークなキャッシュ・タグがミニディレクトリにたとえ存在するとしても、前の読み取りタグ・セットでアクセスされた少なくとも 1 つのタグがミニディレクトリに残っていることを保証するため、ミニディレクトリは 5 つのブロック・タグを記憶する。このようにして、3 線形補間の間 4 つのユニークなキャッシュ・タグの 2 つのセットの間で切り替えが行われても、各セットについて少なくとも 1 つの読み取りキャッシュ・タグが、ミニディレクトリに残るので、4 つのキャッシュ・タグを主ディレクトリと順次比較する必要がない。

【0196】3 線形補間を使用するテクセルのレンダリングの間、キャッシュへの連続的な読み取りは、1 つのマップにおける第 1 のセットの 4 つのテクセルと別のマップにおける第 2 のセットの 4 つのテクセルを読み取る。プリミティブがレンダリングされる際、2 つのマップの各々の範囲内の隣接するテクセルが 1 つおきのサイクル毎にそれぞれアクセスされ、複数のテクセルが、1 つのキャッシュ・ブロック内に一般的に位置づけられる。従って、複数のユニークなタグが 4 つのキャッシュ読み取りタグの各セットに出現するとすれば、各ピクセルのキャッシュ読み取りタグがミニディレクトリ 500 においてヒットする状態を保ちながら多数のピクセルをレンダリングすることができる。もしも 4 つのセットの各々においてただ 1 つのキャッシュ読み取りタグがミニディレクト

りと一致しない場合は、次の4つの読み取りタグセットがミニディレクトリと比較されている間にそのタグを主ディレクトリと比較することができるので、処理性能の低下は派生しない。

【0197】主ディレクトリおよび相対的に小規模のミニディレクトリを含む本発明のキャッシュ・ディレクトリは、テクスチャ・マッピング・ハードウェアでの使用に限定されず、その他の多数のアプリケーションについて使用することができることは理解されるべきであろう。複数のキャッシュ読み取りタグが同時に処理される場合、およびキャッシュ読み取りタグが連続的にアクセスされる以前に使用されたタグと関連している場合、完全連想型キャッシュを実施し、ディレクトリ・タグ比較のコストを低減する上で、本発明のミニキャッシュ・ディレクトリ方式は特に有効である。例えば、1時点でX個のタグを記憶するキャッシュ・メモリについて、N個のキャッシュ読み取りタグが同時にディレクトリ・ブロック・タグと比較されると仮定し、M個のタグを含むミニディレクトリを維持すれば十分である（ただしMはN以上である）。M個のミニディレクトリ・タグの各々は、N個のキャッシュ読み取りタグに対して、1回の読み取り動作において比較される。ミニディレクトリにおいてヒットしないキャッシュ読み取りタグについて、主ディレクトリが順次アクセスされる。そのような読み取りタグは、主ディレクトリ・タグと1回のサイクルで比較される。主ディレクトリのX個のタグの各々がN個の読み取りタグと1回の読み取り動作で比較される場合のシステムにおいて、比較器の観点からのハードウェア節約は、 $(X+M \cdot N)/(X \cdot N)$ という比率に従う。

【0198】このハードウェア節約を達成するために必要とされる処理速度の減少はアプリケーションに依存し、連続的読み取り動作でアクセスされる一連のタグの様態に基づく。各読み取りセットにおいて1つ以上のタグがミニディレクトリとの不一致を起こさなければ、ミスのあったタグと主ディレクトリとの比較は、次のセットの読み取りタグがミニディレクトリと比較されている間に並列的に行うことができるため、処理速度の低下は起きない。

【0199】複数のキャッシュ読み取りタグがミニディレクトリでミスを起こす場合の処理性能の低下を防ぐため使用される上述の比較器526-530に関しては、4つの読み取りタグが同時にアクセスされるので、6つの比較器が使用される。各キャッシュ読み取りタグを他のものと比較するために使われる比較器の数は、同時にアクセスされる読み取りタグの数Nに依存し、1から(N-1)までの整数の和に等しい。

【0200】図22ないし図24のミニディレクトリおよび主ディレクトリを含むキャッシュ・ディレクトリの1つの実施形態が図25に示されている。図25に示されている実施形態は例示の目的のためにすぎず、その他

の形態を実施することができる点は理解されるべきであろう。

【0201】図22のミニディレクトリ・エントリ501-505は、タグ・レジスタ501T-505Tに記憶されるタグ・コンポーネント、および、インデックス・レジスタ501I-505Iに記憶されるインデックス・コンポーネントに分割される。上述のように、キャッシュ・ディレクトリは、処理中のピクセルが対応するMIPマップ位置に最も近い4つのテクセル（すなわちUL、UR、LLおよびLR）に対応する一組の4つの読み取りキャッシュ・タグを受け取る。4つの読み取りタグの各々は、6つのタグ比較器541-546へ送られる。比較器のうちの5つ（すなわち、542-546）は、また、5つのミニディレクトリ・タグ・レジスタ501T-505Tの1つにそれぞれ接続している。例えば、比較器542は、ミニディレクトリ・エントリ1に関するタグ・レジスタ501Tに接続して、ミニディレクトリのそのエントリのタグが読み取りキャッシュ・タグUL、UR、LLまたはLRのいずれかのタグと一致するか否かを標示する出力を生成する。比較器543-546は同様に動作して、読み取りキャッシュ・タグUL、UR、LLまたはLRをミニディレクトリ・エントリ2-エントリ5に対するタグを記憶するタグ・レジスタ502T-505Tとそれぞれ比較する。4つの読み取りキャッシュ・タグの新しいセットの各々が1回のサイクルでミニディレクトリと比較される。そのサイクルの終了時点で、4つのタグUL、UR、LLおよびLRは、レジスタ550-553にそれぞれ記憶される。図25に示されるように、レジスタ550-553の各々は、また、ミニディレクトリ・タグ比較器542-546の出力を受け取る制御回路559に接続する。4つの読み取りキャッシュ・タグの新しい1セットがミニディレクトリ・タグと比較されるサイクルの終了時点で、レジスタ550-553の各々は、その対応するタグ（すなわちUL、UR、LL、LR）がミニディレクトリ・エントリの1つと一致したか否か、一致した場合はどのエントリと一致したかを標示するデータをロードされる。

【0202】上述のように、ただ1つのキャッシュ読み取りタグのミスがミニディレクトリで発生すれば、次のセットの4つのテクセル読み取りタグがミニディレクトリと比較されている間に、そのタグは主ディレクトリと比較される。1つのキャッシュ・ミスがミニディレクトリで発生する場合、ミスを起こしたタグを含むようにミニディレクトリは更新されるので、ミニディレクトリは常に最も最近アクセスされた5つのキャッシュ・タグを反映する。次のセットの4つのテクセル読み取りタグがミニディレクトリと比較されている間にミニディレクトリでミスを起こした読み取りキャッシュ・タグが主ディレクトリと比較されるサイクルの間、ミニディレクトリ・タグ・レジスタ501T-505Tは、前のサイクルでミニ

ディレクトリにおけるミスが発生したキャッシュ・タグを含むようにまだ更新されていない。従って、次のセットの読み込みキャッシュ・タグがミニディレクトリと比較される時、6番目の比較器541を使用して、前のサイクルでミニディレクトリとミスを起こし現在主ディレクトリと比較されているタグと、4つの読取りタグ（UL、UR、LLおよびLR）を比較する。4つのキャッシュ読取りタグ（UL、UR、LLおよびLR）のセットにおける複数のユニークなタグがミニディレクトリのキャッシュ・ミスを発生する場合、主ディレクトリとの複数の比較が発生するため、キャッシュ・ディレクトリを通過するパイプラインが停止させられる。しかし、ただ1つのユニークなタグがミニディレクトリにおけるキャッシュ・ミスを起こすならば、キャッシュ・ディレクトリが各サイクル毎に4つのキャッシュ読取りの新しいセットを受取る後述の方法でパイプラインは続行する。

【0203】上述のように、前のサイクルでミニディレクトリと比較された読取りタグがレジスタ550-553に記憶される。これらのレジスタの出力は、4対1マルチプレクサ555に接続する。このマルチプレクサは、主ディレクトリと比較され、ミニディレクトリが最も最近受け取った読取りキャッシュ・タグで更新されるようにするためサイクルの終了時にミニディレクトリにロードされるべき上記レジスタの1つを選択する。マルチプレクサ555の出力が、また、6番目の比較器541に接続されるので、前のサイクルでミニディレクトリにおけるキャッシュ・ミスを起こしたキャッシュ読取りタグを新しいセットの読取りタグ（UL、UR、LLおよびLR）の各々と比較することができる。比較器542-546との関係で、比較器541は、ミニディレクトリがキャッシュ・ディレクトリによって受け取られた4つのキャッシュ読取りタグの各セットを最も最近受け取った5つの読取りタグと比較することを保証する。

【0204】上述のように、マルチプレクサ555から出力されるキャッシュ読取りタグが、主ディレクトリと比較されるサイクルの終了時点で、ミニディレクトリ・タグ・レジスタ501Tないし505Tの1つにロードされる。このようにして、ミニディレクトリは、最も最近アクセスされたキャッシュ・タグを含むように更新される。マルチプレクサ555から出力される新しいキャッシュ・タグでどのエントリを更新するかは、以下に記述の置き換え方式に基づいて行われる。

【0205】図24に関連して記述された6つの比較器セット526-532が、図25においては便宜上単一の比較器ブロックとして示されている。これらの比較器の出力は、各々が制御回路559に送られる比較器541-546の出力と共に、いくつかの機能を実行する。ミニディレクトリに対するキャッシュ・ミスが発生すると、制御回路559は、ミニディレクトリの中のどのエントリが新しい読取りキャッシュ・タグと置き換えられ

るべきかを決定する。制御回路559は、ミニディレクトリに対して比較されている新しく受け取った4つの読取りキャッシュ・タグの1つ、または主ディレクトリと比較された最後の読取りキャッシュ・タグによってヒットされたいかなるエントリも置き換えることはせず、ミニディレクトリに維持される最高優先度をこれらのエントリに割り当てる。加えて、制御回路559は、4つの読取りタグの先行セットによるヒットがあったミニディレクトリ・エントリに関する状態情報を記憶し、ミニディレクトリに維持される次に高い優先度をそれらのエントリに割り当てる。残りのエントリには低い優先度が割り当てられる。

【0206】制御回路559は、少なくとも1つのエントリを含む最下位優先度グループにあるエントリを置き換えるため選択する。このように、ミニディレクトリに対して比較されている新しく受け取った4つの読取りキャッシュ・タグの1つによってヒットされず、主ディレクトリに対して比較された最後の読取りキャッシュ・タグでなく、4つの読取りタグの先行セットに含まれていなかった低位優先度グループに少なくとも1つのエントリがあれば、その低位優先度グループの中のエントリの1つが置き換えるため選択される。しかし、低位優先度グループにエントリがなければ、最優先グループのエントリ（すなわち新しく受け取った4つの読取りキャッシュ・タグの1つおよび主ディレクトリと比較された最後の読取りキャッシュ・タグによってヒットされたエントリ）を除く高位優先度グループのエントリから置き換えるためのエントリが選択される。

【0207】使用可能な最下位優先度ミニディレクトリ・エントリのグループが識別されると、5つのミニディレクトリ・エントリの各々が1度に1つずつ置き換えられるサイクルをとる置き換え方式に従って、そのグループ内のどのエントリを置き換えるべきか決定される。この決定を行う方法は多数ある。本発明の1つの実施形態においては、5つのミニディレクトリ・エントリに1ないし5というラベルが付けられる。置き換えられるべきエントリを最下位優先度グループから選択する場合、先ず当該グループにない最も高い番号のエントリを識別し、次に、当該グループ内で次に最も高い番号を持つエントリを置き換え用を選択する。エントリ5が最下位優先度グループにない場合、置き換え方式は1にもどって、エントリ1が次の最高の番号を持つエントリとみなされる。この置き換え方式によって、制御回路559は、ミニディレクトリ・エントリが1サイクルに1つずつ置き換えられるようにサイクルを進め、選択されたミニディレクトリ・タグ・レジスタ501T-505Tへの書き込みを制御する。

【0208】制御回路559は、また、比較器541-546の出力をデコードして、読取りタグがミニディレクトリの1つのエントリと一致したか否か、一致したと

10

20

30

40

50

すればどのエン트리と一致したかを標示するデータを4つの読取りタグ(UL、UR、LLおよびLR)の各々毎に生成する。このデータは、4つの読取りタグ(UL、UR、LLおよびLR)の各々に対応するレジスタ550-553に記憶される。例えば、読取りタグULがミニディレクトリ・エン트리3と一致すれば、制御回路559によってデコードされたデータは、ULレジスタ550に記憶され、読取りタグがミニディレクトリ・エン트리3と一致したことを標示する。後述するように、このデータは、キャッシュ・ディレクトリ・パイプラインを通過し、ULテクセルに関するブロック・インデックスが、ミニディレクトリ・エン트리3に関するブロック・インデックスを保持するレジスタ503Iに記憶されていることを標示する。

【0209】読み取りタグセットUL、UR、LLおよびLRのただ1つのユニークなタグがミニディレクトリと一致しない場合、対応するテクスチャ・データに関するブロック・インデックスがミニディレクトリにないことを標示するデータが、その読取りタグを記憶するレジスタ550-553の各々にロードされる。次のサイクルの間、不一致のタグを記憶するレジスタ550-553の1つの出力が、主ディレクトリ520と比較され、読取りタグに対するブロック・インデックスが、主ディレクトリから、主ディレクトリ・ブロック・インデックスを記憶するレジスタ561にロードされる。また、ブロック・インデックスがミニディレクトリのエントリに対応しないことを標示するデータが、マルチプレクサ555の出力から、入力562経由で、レジスタ561に記憶される。

【0210】上述のように、4つのテクセルが同時にアクセスされることができるようキャッシュ・メモリは4つのインターリーブA-Dを含む。4つのテクセル読取りタグUL、UR、LLおよびLRのセットは、いかなる形態でもインターリーブA-Dに対応することができる。レジスタ550-553に記憶され、どのミニディレクトリ・エントリがテクセルUL、UR、LLおよびLRの各々に対応するかを標示するデータは、テクセルUL、UR、LLおよびLRの各々をその対応するインターリーブA-Dに関連づけるように制御される軸シフタ(barrel shifter)563を通過する。軸シフタの出力は、それぞれインターリーブA-Dに対応するインターリーブ・インデックス制御レジスタ565-568にロードされる。インターリーブ・インデックス制御レジスタの各々は、インターリーブに関するブロック・インデックスを記憶するミニディレクトリ・エントリを識別する。ただ1つのユニークな読み取りキャッシュ・タグだけがミニディレクトリと一致しない場合、レジスタ550-553からの出力のシフトおよびレジスタ565-568の書き込みが、主ディレクトリ520へのアクセスと並列して行われる。

【0211】上述のように、レジスタ565-568にロードされるデータは、対応するインターリーブに関するブロック・インデックスを記憶するミニディレクトリ・エントリを識別する。このデータは、ミニディレクトリ・インデックス・レジスタ501I-505Iの1つおよび主ディレクトリ・ブロック・インデックス・レジスタ561から、各インターリーブに対する対応するブロック・インデックスを選択する複数のインターリーブ・インデックス・マルチプレクサ571を制御するために使用される。複数のインターリーブ・インデックス・マルチプレクサ571は、4つの独立した6対1マルチプレクサを表す。1つのマルチプレクサは、各インターリーブに対応して、5つのミニディレクトリ・インデックス・レジスタ501I-505Iおよび主ディレクトリ・ブロック・インデックス・レジスタ561から1つを選択する。各インターリーブ・インデックス・マルチプレクサは、同じインターリーブに対応し、どのミニディレクトリ・エントリが該インターリーブに対するブロック・インデックスを記憶しているかを識別するレジスタ565-568の1つによって制御される。インターリーブに対するブロック・インデックスがミニディレクトリ・エントリにないことをこのデータが標示する場合、対応するマルチプレクサが、ミニディレクトリ不一致に引き続いて主ディレクトリから読み取られたブロック・インデックスを記憶する主ディレクトリ・ブロック・インデックス・レジスタ561から提供されるインデックスを選択する。インターリーブA-Dの各々に対するブロック・インデックスは、ライン580-583上に送出され、上述の方法でキャッシュSDRAMをアドレス指定するために使用される。

【0212】上述のように、読み取りキャッシュ・タグUL、UR、LLおよびLRのセットの複数タグがミニディレクトリと一致せずしかした1つのユニークなキャッシュ・タグだけを含む場合、その読み取りタグに関するブロック・インデックスを提供するため、主ディレクトリ520は1回だけアクセスされる。このプロセスも、4つの読取りタグのどの2つが一致するかを識別する比較器526-532の出力を使用する制御回路559によって制御される。4つの読取りタグのセットの2つ以上が、同じキャッシュ・タグを含むミニディレクトリと一致しない場合、対応するレジスタ550-553の各々は、ブロック・インデックスがミニディレクトリ・エントリに含まれないことを標示するように制御回路559によってセットされる。このようにして、読み取りタグに対応するデータが、インターリーブ・インデックス・レジスタ565-568を通過する時、各々がその対応するインターリーブ・インデックス・マルチプレクサ571に送られるように主ディレクトリ・ブロック・インデックス制御レジスタ561を選択する。

【0213】制御回路559は、また、読取りタグ・レ

レジスタ550-553のどれが主ディレクトリに対して比較されるべきかを制御するディレクトリ制御レジスタ573をセットする。レジスタ573は、主ディレクトリに対して1時点に比較されるべきレジスタ550-553の1つを選択するようにマルチプレクサ555を制御する。読取りタグUL、UR、LLおよびLRの1つがミニディレクトリと一致せずしかし共通タグを共有しない場合、制御レジスタ573は、レジスタの中の1つだけが主ディレクトリに対して比較されるべきであることを標示するようにセットされる。このようにして、4つの読取りキャッシュ・タグのセットがミニディレクトリとの不一致を起こすただ1つのユニークなタグだけを含む場合、主ディレクトリは一度だけアクセスされる。

【0214】4つの読取りキャッシュ・タグUL、UR、LLおよびLRのセットがミニディレクトリとの不一致を起こす2つ以上のユニークなタグを含む場合、キャッシュ・ディレクトリ・パイプラインを通過する上記の流れが変更され、キャッシュ・ディレクトリは、ビジー(busy)となって読取りタグの新しいセットを次のサイクルで受け取らない。ディレクトリがビジー(busy)であることを標示することによって、ミニディレクトリと一致しなかった読取りタグを含むレジスタ550-553の各々が主ディレクトリに対して比較されることができ、新しい読取りタグで上書きされない。更に、ディレクトリ・パイプラインを通過する流れが変更されるため、ミニディレクトリと一致しなかった読取りタグの各々について主ディレクトリがアクセスされ、それらに対応するブロック・インデックスが、主ディレクトリからレジスタ501I-505Iまたは561の1つにロードされることができる。読取りキャッシュ・タグUL、UR、LLおよびLRのセットに対するブロック・インデックスのすべてが主ディレクトリから読み取られるか、あるいは既にミニディレクトリに存在するようになるまで、パイプラインは、レジスタ550-553のいずれのデータも軸シフト563を通過させないように構成される。このように、テクセルUL、UR、LLおよびLRのセットは、グループとして対応するインターリーブに関連づけられる。

【0215】読取りタグの1セット中の複数のユニークなタグがミニディレクトリと一致しない場合、不一致のタグは順次処理される。(タグのセットがミニディレクトリと比較される)第1のサイクルの間、制御回路559は、ミニディレクトリの中のどのエントリが第1の不一致読取りタグによって置き換えられるべきかを判断し、対応するレジスタ550-553には、そのブロック・インデックスがそのミニディレクトリ・エントリに記憶されることを標示するデータがロードされる。最初に処理された不一致タグを記憶するレジスタ550-553の出力が、第2のサイクルの間に、主ディレクトリ520と比較される時、主ディレクトリ・ブロック・イ

ンデックス・レジスタ561は、ミニディレクトリ・インデックス・レジスタ501I-505Iのどれが置き換えられるべきかを標示するデータを用いて更新される。第3のサイクルの間、対応するブロック・インデックスは、レジスタ561から、置き換えのため選択されたミニディレクトリ・エントリに対応するレジスタ501I-505Iにロードされる。

【0216】ミニディレクトリと一致しなかった後続のユニークなタグの各々は、それが処理されるべき最後のタグとなるまで同じ方法で処理される。キャッシュ・ディレクトリを通過して処理される最後の不一致タグは、あたかも読取りタグのセット中でミニディレクトリと一致しない唯一のユニークなタグであるかのように取り扱われる。最後の不一致タグの処理を開始する時、ディレクトリは、読取りタグの新しいセットを受け取ることができるようにするため、ビジー(busy)であることを標示する信号をオフにセットする。

【0217】最後に処理される不一致タグに関して、制御回路559は、その対応するレジスタ550-553に、該タグに対するブロック・インデックスがミニディレクトリ・エントリに記憶されていないことを標示するデータをロードする。これは、すべての読取りタグがミニディレクトリと比較される第1のサイクルの間に、またはその他の不一致タグの処理と並列して、実行される。最後の不一致タグが主ディレクトリと比較されるサイクルの間、レジスタ550-553のデータが、軸シフト563を通過して、インターリーブ制御レジスタ565-568にロードされ、不一致タグに関するブロック・インデックスは、主ディレクトリから、主ディレクトリ・ブロック・インデックス・レジスタ561にロードされる。最後に、ディレクトリの最終パイプライン段階において、インターリーブ・インデックス制御レジスタ565-568の出力を使用して、それらの対応するインターリーブ・インデックス・マルチプレクサ571を制御することによって、最後に処理された不一致タグに関するインデックスが主ディレクトリ・ブロック・インデックス・レジスタ561から送られ、該セット中の他の読取りタグの各々に関するブロック・インデックスが、その対応するミニディレクトリ・インデックスレジスタ501I-505Iから渡される。最後に処理された不一致タグに関するブロック・インデックスを主ディレクトリ・ブロック・インデックス・レジスタ561からアクセスすることによって、サイクルは、このタグに関するブロック・インデックスがそのミニディレクトリ・インデックス・レジスタにロードされるのを待たないので、1サイクルが節約される点理解されるべきであろう。

【0218】以上、本発明の少なくとも1つ実施形態を記述したが、当業者にとって種々の変更、修正および改良を行うことは容易であろう。そのような変更、修正お

よび改良は本発明の精神および対象範囲内にあるものと意図されている。従って、上記記述は、例示のためのものにすぎず、本発明をそれに限定するものとして意図されていない。

【0219】本発明には、例として次のような実施様態が含まれる。

(1) テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、個別にアクセス可能な少なくとも第1および第2のメモリ領域にテクスチャ・データを割り当てる方法であって、少なくとも一連のテクスチャMIPマップを、等しいサイズの複数のデータ・ブロックに分割するステップと、上記少なくとも一連のテクスチャMIPマップの中の1つおきのMIPマップに含まれる共通のテクスチャ・データを含む第1のブロックを上記第1のメモリ領域に記憶するステップと、上記第1のブロックに含まれるテクスチャ・データおよび上記1つおきのMIPマップに隣接するMIPマップに共通に含まれる共通のテクスチャ・データを含む第2のブロックを上記第2のメモリ領域に記憶するステップと、を含む方法。

(2) 上記分割するステップが、上記少なくとも一連のテクスチャ・マップの各マップを少なくとも2つの部分に分割するステップを含む、上記(1)に記載の方法。

(3) 上記分割するステップが、更に、上記マップ部分を複数の等サイズのブロックに割り当てるステップを含む、上記(2)に記載の方法。

【0220】(4) テクスチャ・マッピング・コンピュータ・グラフィックス・システムにおいて、少なくとも一連のテクスチャMIPマップを含むテクスチャ・データを割り当て、記憶する方法であって、上記少なくとも一連のMIPマップの各マップを少なくとも2つのマップ部分に分割するステップと、1つのブロック内に割り当てられるマップの部分が1つのブロックのサイズより小さくなるように各マップの部分を複数の等サイズのデータ・ブロックに割り当てるステップと、上記複数のデータ・ブロックを該システムの主メモリ内に記憶するステップと、1回に少なくとも1ブロックずつ上記主メモリから該システムのローカル・メモリへ上記ブロックをダウンロードするステップと、を含む方法。

(5) 上記分割するステップが、各マップを4つの象限に分割するステップを含む、上記(4)に記載の方法。

(6) ブロックに記憶された特定のマップ、ブロックに記憶されたマップ部分、およびブロックに記憶されたマップのテクスチャを識別するため、各ブロックにブロック・タグを割り当てるステップを更に含む、上記(4)に記載の方法。

(7) 各ブロックを複数のサブテクスチャへ小分割して、サブテクスチャをブロックの未使用空間内に割り当てるステップを更に含む、上記(4)に記載の方法。

【0221】(8) テクスチャ・マッピング・コンピュ

ータ・グラフィックス・システムにおいて、少なくとも一連のテクスチャMIPマップを含むテクスチャ・データのブロックを記憶するためのメモリであって、上記少なくとも一連のテクスチャMIPマップの中の1つおきのMIPマップに含まれる共通のテクスチャ・データを含む少なくとも第1のブロックを記憶する第1のメモリ領域と、上記第1のメモリ領域とは別個にアクセスすることが可能であって、上記第1のブロックに含まれるテクスチャ・データおよび上記1つおきのMIPマップに隣接するMIPマップに共通に含まれる共通テクスチャ・データを含む少なくとも第2のブロックを記憶する第2のメモリ領域と、を備えるメモリ。

(9) 少なくとも1つのSDRAMを含む上記(8)に記載のメモリ。

(10) 上記第1および第2のメモリ領域が、少なくとも1つのSDRAMの第1および第2のバンクをそれぞれ含む、上記(9)に記載のメモリ。

【0222】

【発明の効果】本発明に従うテクスチャ・マッピング・ハードウェアへのテクスチャ・データのダウンロードが、プリミティブ伝送経路と異なる経路を使用して実施され、従って3次元プリミティブ・パイプラインのフラッシングを必要としないので、システムの帯域幅および処理能力が向上する。また、本発明の1つの実施形態において、特定のプリミティブ・レンダリング・タスクが複数のプリミティブに関して並列的に実行されるように、テクスチャ・マッピング基板およびフレーム・バッファ基板に関する諸ハードウェアが反復配置されるため、システムの帯域幅が拡大される。

【0223】更にまた、本発明に従うキャッシュ・メモリ内のSDRAMチップの各々は、同時に別々の活動ページ（すなわち、共通の行アドレスを持つメモリ位置グループ）を維持することができる2つの等しいサイズのバンクに内部的に分割されるため、従来技術のDRAMの場合に起きるような2つの異なるページ（すなわち2つの異なる行アドレス）からデータを取り出すことに付随する再ページングの負荷を伴うことなく、SDRAMチップの2つのバンク内の異なるページにあるデータを連続的読取りサイクルでアクセスすることができるという効果を持つ。

【0224】また、本発明の1つの実施形態において、多数の近接した読取りサイクル内で共通のテクセル・データが多数回アクセスされる場合、キャッシュは、最初の読取りについてののみアクセスされるだけで、後続の読取りの各々についてはキャッシュ読取りサイクルが節約され、これによって、必要とされるキャッシュ・アクセス数が減り、システムの帯域幅が増大する。

【0225】加えて、本発明に従って、キャッシュ・ディレクトリを主ディレクトリおよびミニディレクトリに分割構成することによって、同一のタグについて主ディ

レクトリをアクセスする場合複数サイクルを伴う必要がなくなるので、複数のキャッシュ読み取りタグがミニディレクトリで一致ミスを起こす場合のシステム帯域幅に対する影響を最小限にとどめることができ、高いシステム帯域幅を達成する一方キャッシュ・ディレクトリを実施するハードウェアの数量を比較的少なくすることができる。

【0226】このように、本発明は、種々の局面において、コンピュータ・グラフィックス・システムなどのデータ処理システムの処理能力を向上させる効果を奏する。

【図面の簡単な説明】

【図1】一組のテクスチャMIPマップの例を示す図である。

【図2】本発明のコンピュータ・グラフィックス・システム全体の1つの実施形態のブロック図である。

【図3】本発明のコンピュータ・グラフィックス・システム全体の別の1つの実施形態のブロック図である。

【図4】本発明のテクスチャ・マッピング・ハードウェアのブロック図である。

【図5】本発明のテクスチャ・マッピング・ハードウェアのパラメータ補間回路エレメントのブロック図である。

【図6】本発明のキャッシュ・メモリおよびテクスチャ・マッピング・ハードウェアの一部を示すブロック図である。

【図7】テクスチャ・データのブロックが、本発明のキャッシュ・メモリの4つのインターリーブ形態の利点を生かすように構成される様態を示す図である。

【図8】本発明のキャッシュ・メモリを形成するメモリ・チップの構成を示すブロック図である。

【図9】本発明のテクスチャ・マッピング・ハードウェアの一部を示すブロック図である。

【図10】本発明のテクスチャ・マッピング方式に従って、ピクセルの各々について隣接するMIPマップからアクセスされるテクセルの例を示す図である。

【図11】テクスチャ・マッピング・ハードウェア・バッファおよび図10の例に従う関連データのエントリを示す図である。

【図12】本発明のテクスチャ・マッピング・ハードウェアによって使われる回路のブロック図である。

【図13】一組のテクスチャMIPマップの例を示す図である。

【図14】本発明のメモリ記憶方式に従って図13の例のMIPマップがメモリに記憶される形態を示す図である。

【図15】本発明のメモリ記憶方式に従ってMIPマッ *

* プが細分化される形態を示すMIPマップのブロック図である。

【図16】本発明のメモリ記憶方式に従ってMIPマップが更に細分化される形態を示す図15のMIPマップ部分のブロック図である。

【図17】キャッシュ・ブロック・タグが生成される形態を示す図である。

【図18】補間されたテクセルを基に対応するテクスチャ・データ・ブロックを持つテクセル・アドレスを決定するプロセスを示す流れ図である。

【図19】キャッシュ・ミスが発生する時置き換えられるべきキャッシュ・ブロックを決定するプロセスを表す流れ図である。

【図20】テクスチャ・マッピング・チップにおいて提供されるテクセル・ポート・レジスタを示す図である。

【図21】ホスト・コンピュータにおいてキャッシュ・ミス割り込みを処理するプロセスを示す流れ図である。

【図22】キャッシュのミニディレクトリのブロック図である。

【図23】キャッシュの主ディレクトリのブロック図である。

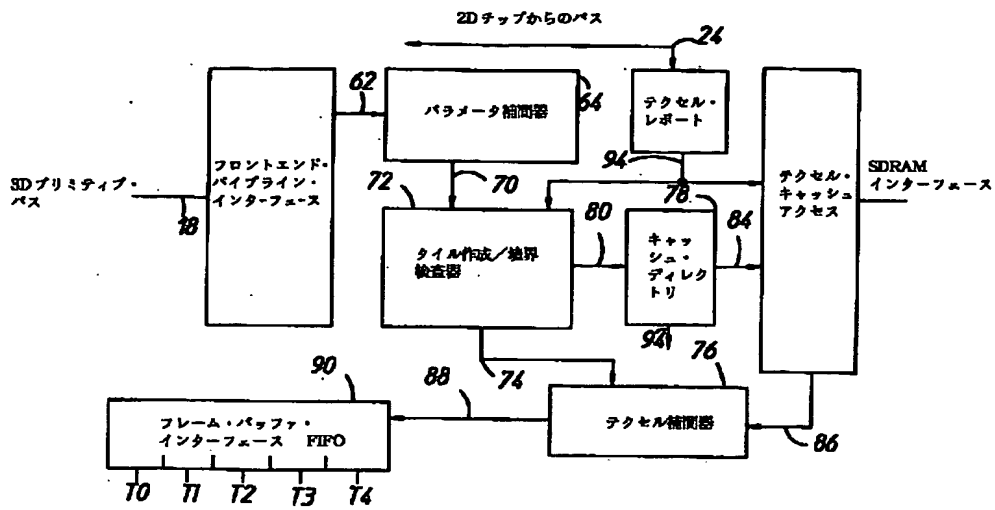
【図24】キャッシュ読み取りタグがミニディレクトリと一致しない場合の処理能力低下を防ぐために用意される一連の比較器のブロック図である。

【図25】本発明のキャッシュ・ディレクトリの1つの実施形態を表すブロック図である。

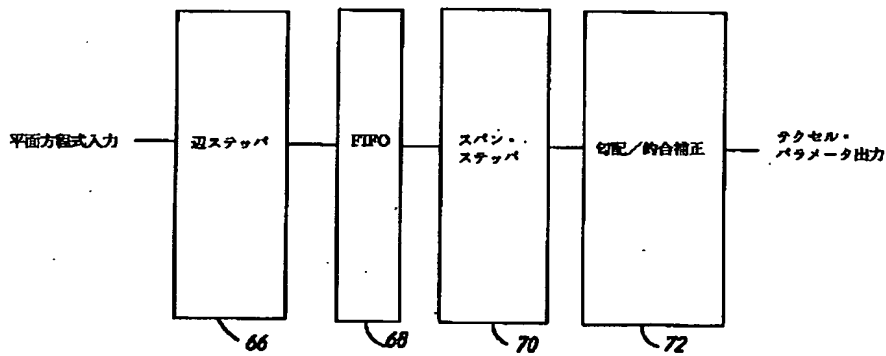
【符号の説明】

- 10 フロントエンド基板
- 12 テクスチャ・マッピング基板
- 14 フレーム・バッファ基板
- 15 ホスト・コンピュータ
- 46 テクスチャ・マッピング・チップ
- 48 ローカル・メモリまたはキャッシュ・メモリ
- 100 基本マップ
- 102、104、108 MIPマップ
- 110、112、130 テクセル
- 132 加重平均テクセル
- 204A、204B、204C、204D インターリーブ
- 206A、206B、206C、206D キャッシュ・アクセス・コマンドFIFO
- 214A0、214A1、214B0、214B1、214C0、214C1、214D0、214D1 テクセル・データFIFO
- 216 テクセル補間回路コマンドFIFO
- 555 4対1マルチプレクサ

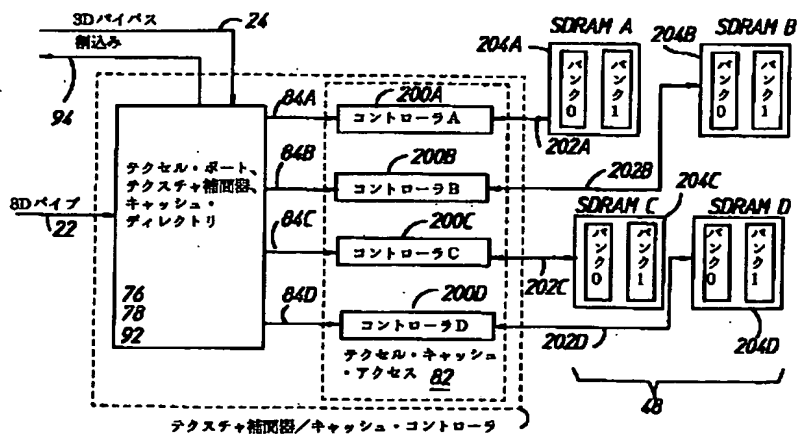
【図 4】



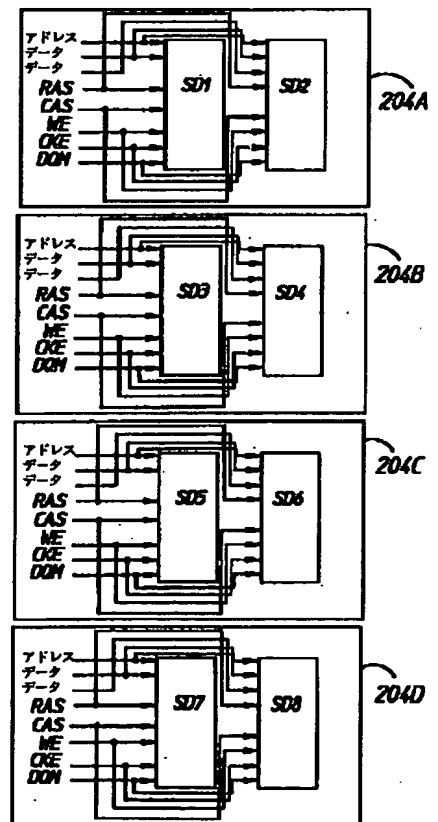
【図 5】



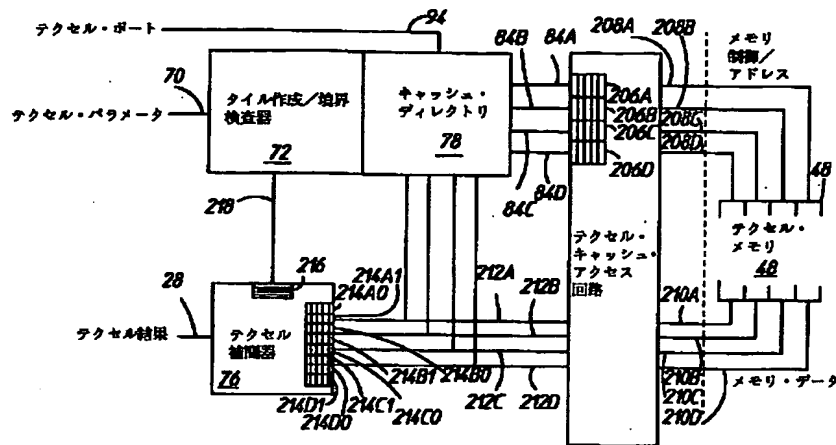
【図 6】



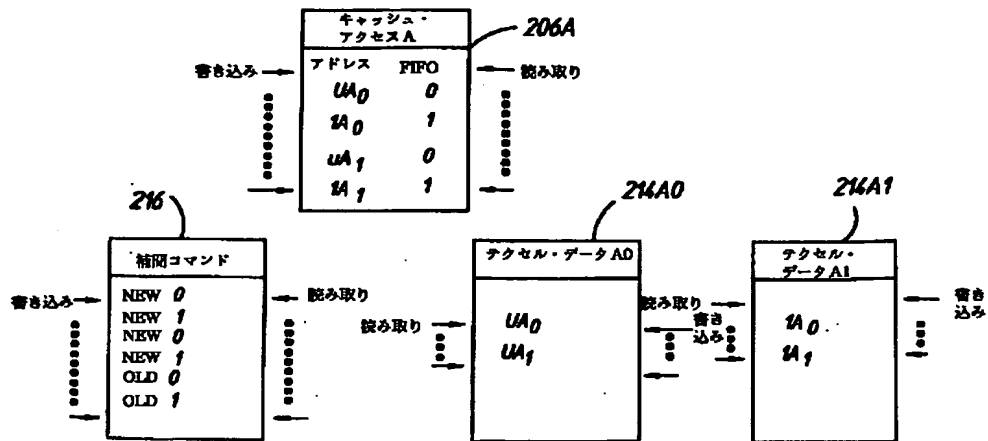
【図 8】



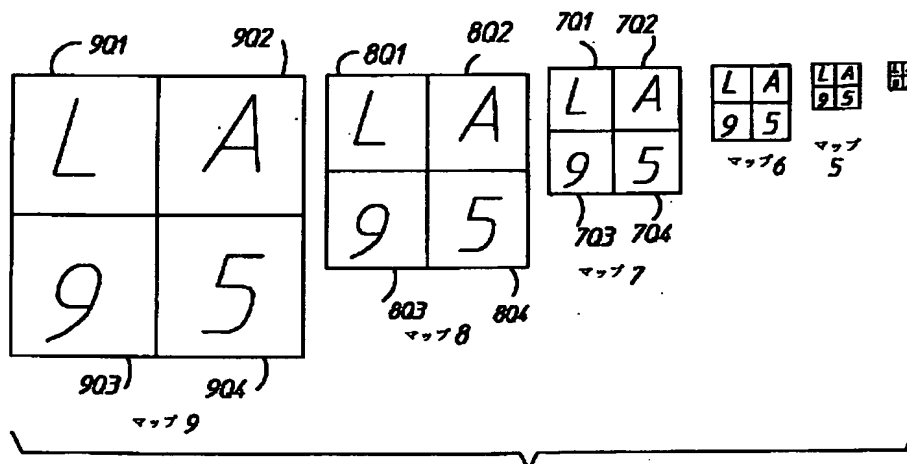
【図 9】



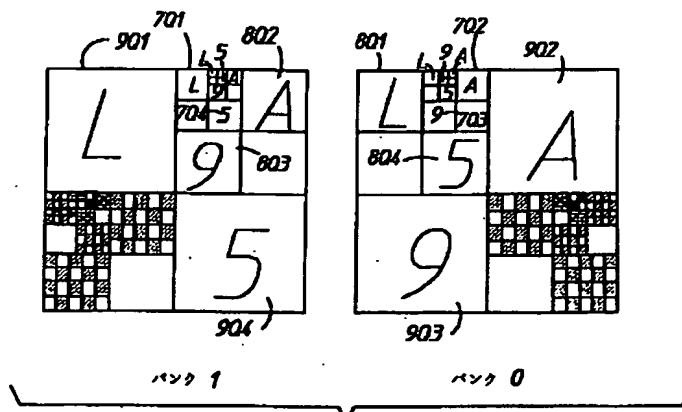
【図 1 1】



【図 1 3】



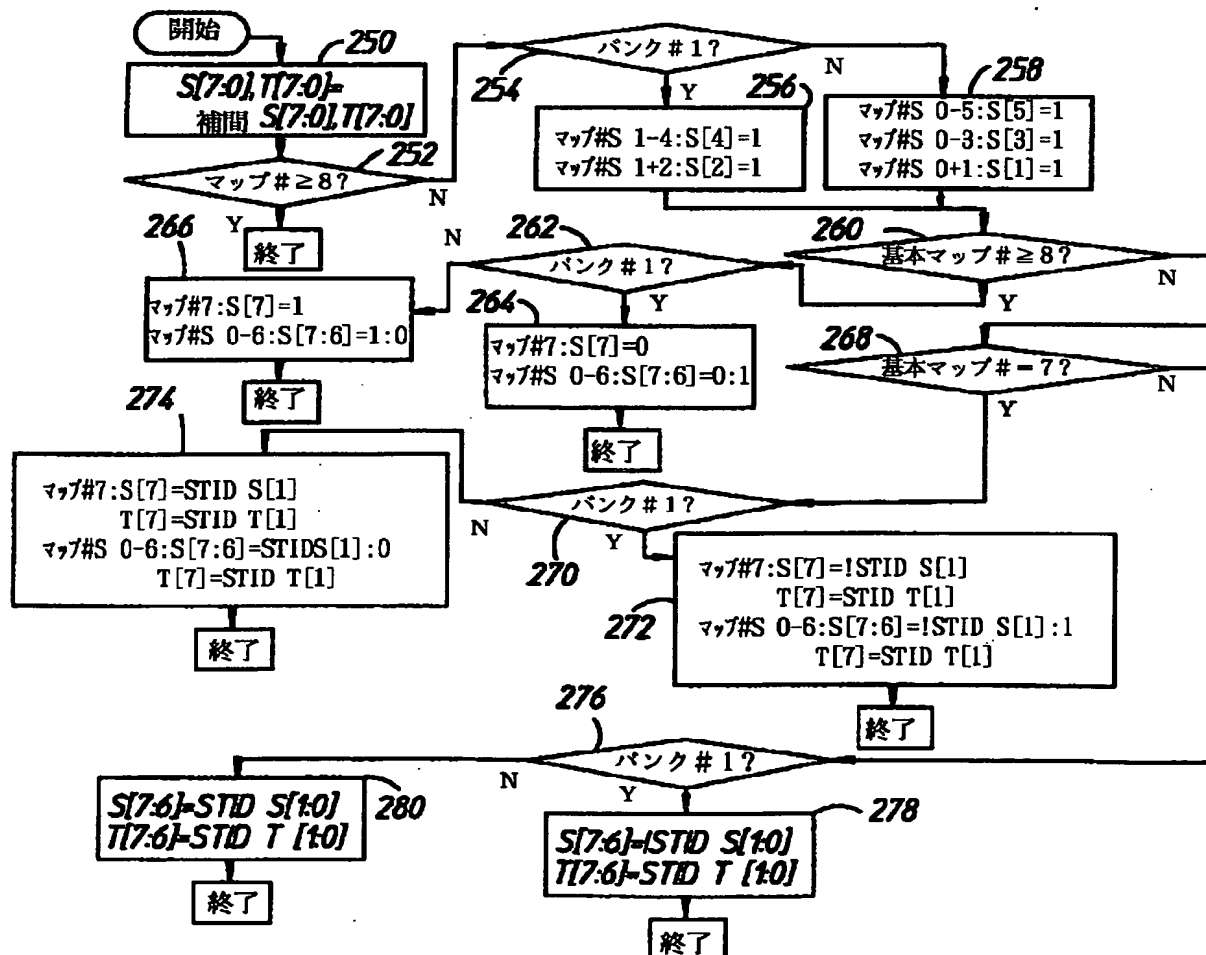
【図 1 4】



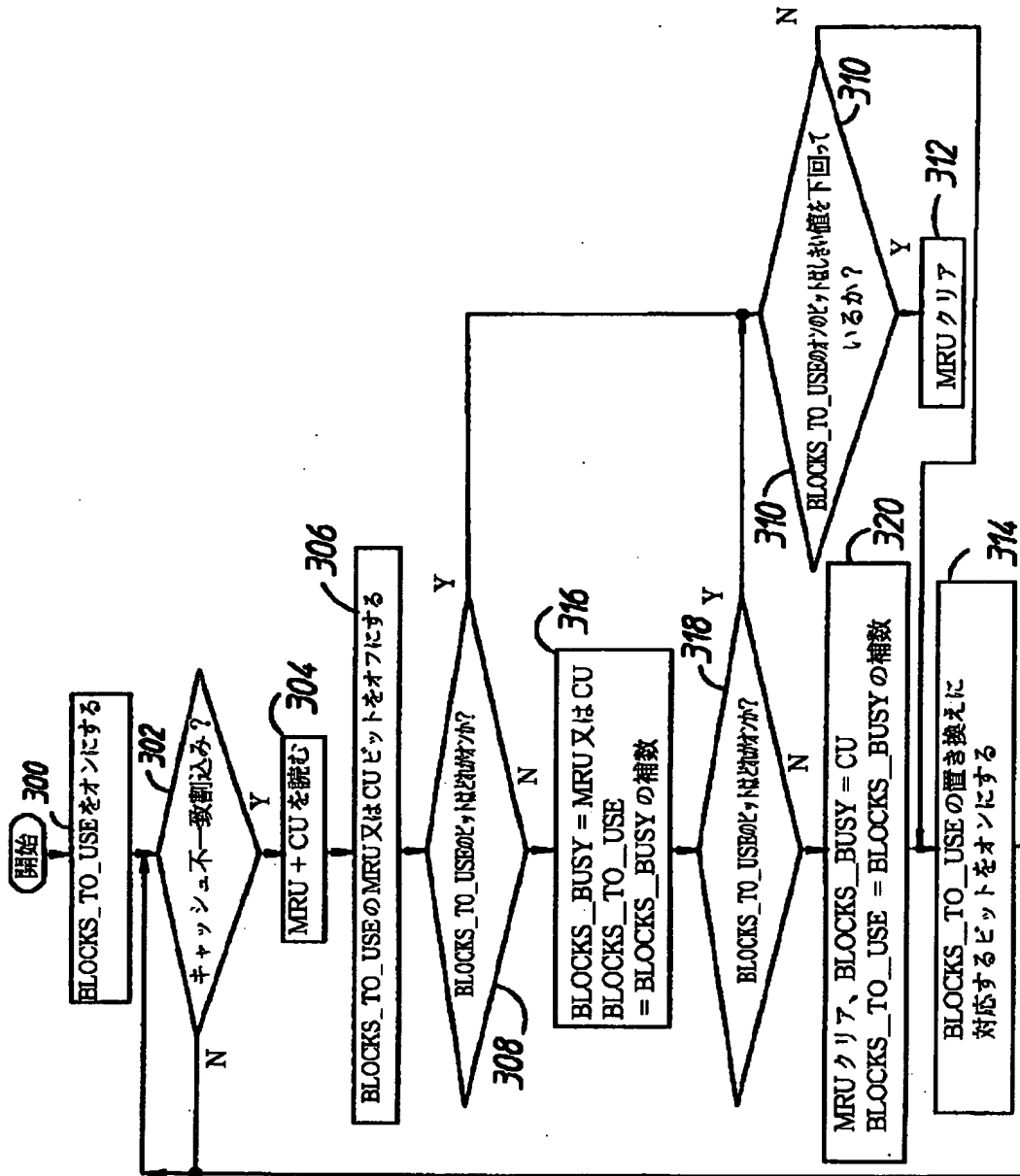
【図 17】

| ブロック・タグ [22:0] - {テクスチャID [1:0], マップ・ビット, T[14:8], S[14:8]} | | | | | | | | | | | | | | | | |
|---|------------|-------------|----|----|----|----|-----|-----|-------------|-----|-----|-----|-----|-----|-----|---------|
| マップ 番号 | マップ ビット | Tパラメータ上位ビット | | | | | | | Sパラメータ上位ビット | | | | | | | バンク |
| 5 | 0 | 14 | 13 | 12 | 11 | 10 | 109 | 108 | s14 | s13 | s12 | s11 | s10 | s09 | s08 | 114^s14 |
| 14 | 1 | 0 | 13 | 12 | 11 | 10 | 109 | 108 | 0 | s13 | s12 | s11 | s10 | s09 | s08 | 113^s13 |
| 13 | 1 | 1 | 0 | 12 | 11 | 10 | 109 | 108 | 0 | 0 | s12 | s11 | s10 | s09 | s08 | 112^s12 |
| 12 | 1 | 1 | 1 | 0 | 11 | 10 | 109 | 108 | 0 | 0 | 0 | s11 | s10 | s09 | s08 | 111^s11 |
| 11 | 1 | 1 | 1 | 1 | 0 | 10 | 109 | 108 | 0 | 0 | 0 | s11 | s10 | s09 | s08 | 110^s10 |
| 10 | 1 | 1 | 1 | 1 | 1 | 0 | 109 | 108 | 0 | 0 | 0 | 0 | 0 | s09 | s08 | 109^s9 |
| 9 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 108 | 0 | 0 | 0 | 0 | 0 | 0 | s08 | 108^s8 |
| ≤8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | タグ [7] |
| ≤8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | タグ [7] |

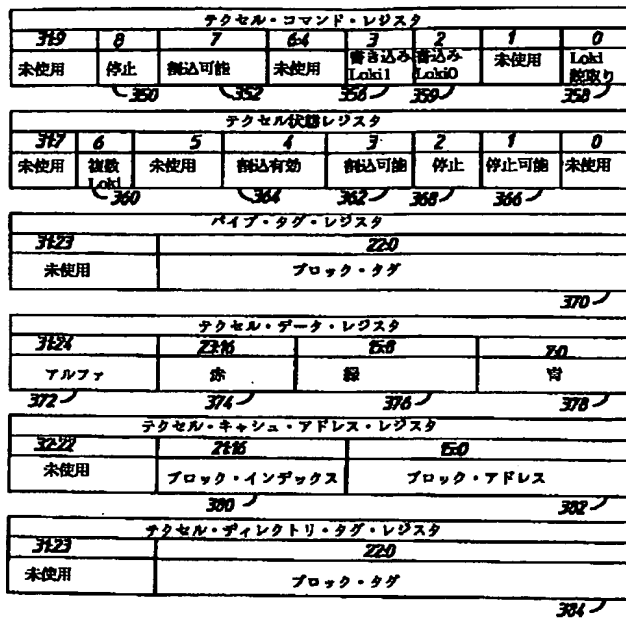
【図 18】



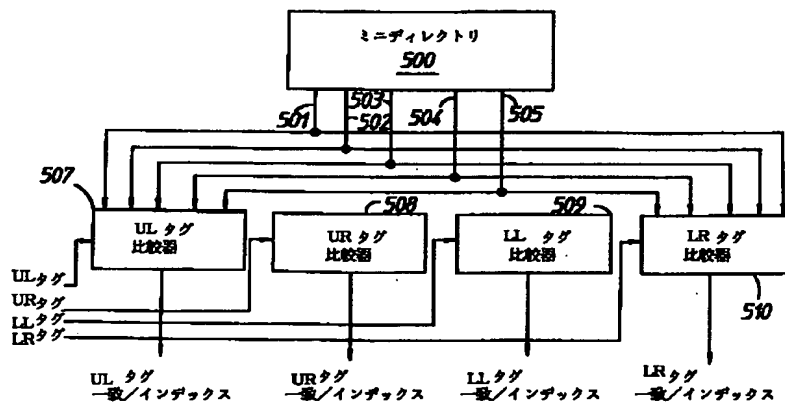
【図 19】



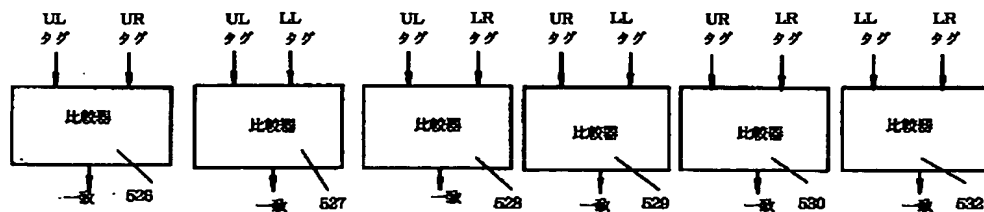
【図 20】



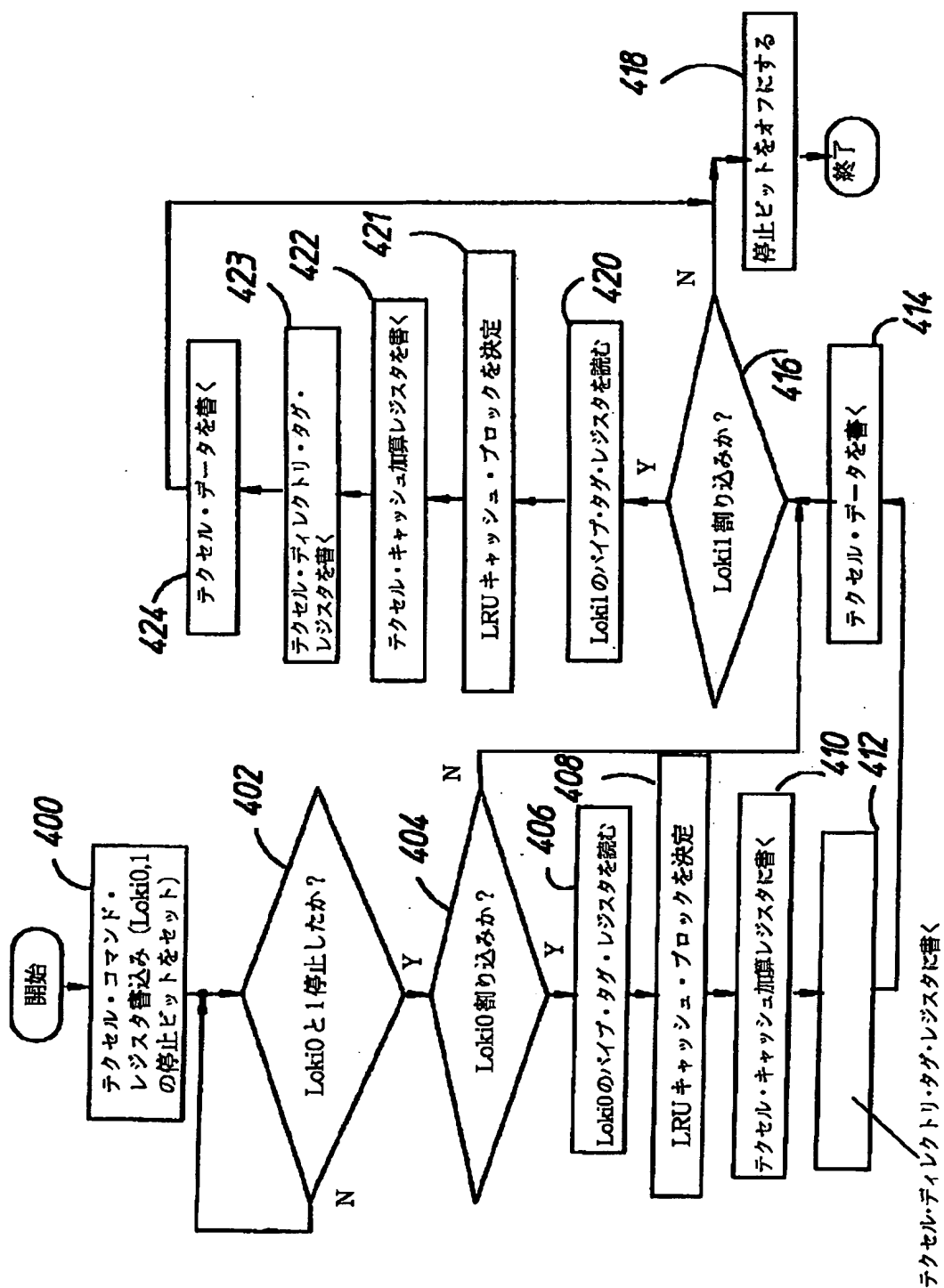
【図 22】



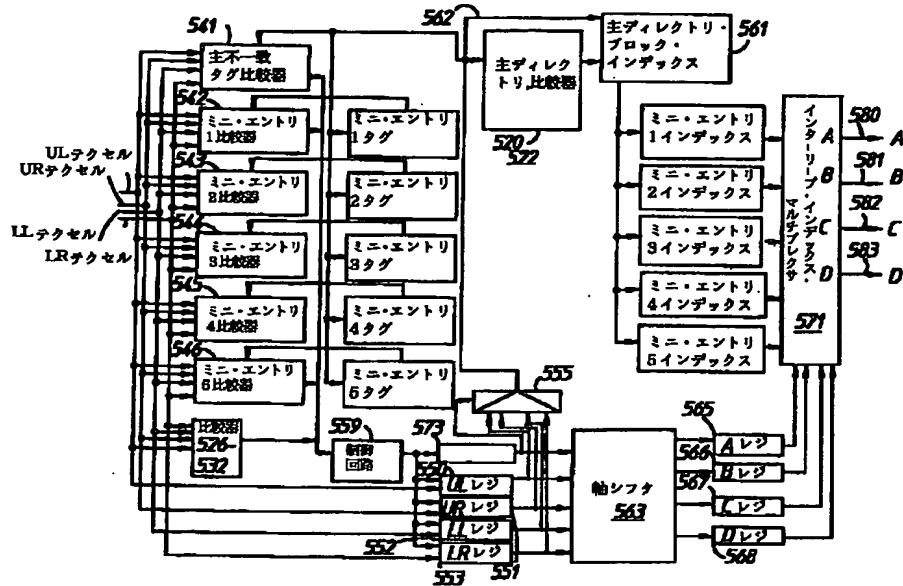
【図 24】



【図 21】



【図25】



**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.